Vol.2015-SE-187 No.15
2015/3/12

クラス図を用いた基礎的概念モデリングにおける誤り分析に基づく初学者向け誤り自動検出機能の開発

増元健人[†1]　香山瑞恵[†2]　小形真平[†2]　橋本昌巳[†2]

本研究の目的は，初学者を対象としたクラス図による概念モデリングの学習支援方法を探究することである．これまでに，初学者によるクラス図を用いた概念モデリングは，手書きによる学習方法を本研究では用いてきた．そこで見られた誤りを元に，クラス図を用いた課題における初学者の誤りの傾向を分析し，17 の誤りパタンとして整理した．また，誤りパタンの内，自動で検出が可能な 9 項目をモデルエディタに実装する事で，初学者の学習支援を行った．本稿では，誤りの自動検出機能をモデルエディタに実装した成果について報告する．

Developing an Error Detecting Tool based on Quantitative Error Analyses for Novices Learning to Create Error-free Simple Class Diagrams

KENTO MASUMOTO[†1]　MIZUE KAYAMA[†2]
SINPEI OGATA[†2]　MASAMI HASHIMOTO[†2]

The purpose of this study is to explore educational methods for conceptual modeling for novices. In this research, the subjects are freshmen in university enrolled in various programs. Typical errors in class diagrams made by four types of novice learners are not easily detected. Therefore, we collected and analyzed some common errors, then developed a simple error detecting function as the ASTAR plugin tool for novices based on these errors. In this paper, we show the results of our analyses of class diagram errors. After that we discuss our basic tool to teach conceptual modeling to novices effectively.

1. はじめに

モデリングとは，対象世界の概念構造を描く活動である[1]．情報システムの世界でモデリングは，データベース設計やソフトウェア設計などに利用されている．さらに，近年では各種センサによる計測を行う組込デバイスのハードウェア設計などにも利用される．モデリングを導入することで，顧客の要求を正しく開発者に伝えることができる．そのため，要求に対する実現機能の正確さが増し，顧客の満足度が向上すると考えられる．このことから，モデリングはシステム開発において重要な位置を占めるものであり，モデリング教育は情報系の高等教育機関において期待の大きな科目である[2]．

しかし，モデリングの方法や良いモデルの判断基準などは言語表現が難しく，初学者にとっては学習に困難を感じやすい科目であるといわれる．モデリング学習を行う学生の理解度を向上させることは，情報系学部の重要な課題点であると考える．

また，モデリングの際には「モデル図やアルゴリズムを設計するための概念形成能力」，「ソフトウェア・ハードウェア要求から必要な情報を読み取る要求分析能力」，「対象の特徴や本来の姿をより明確に示す抽象化能力」などが求められる[3]．しかし，これらの能力等に関する初学者向けの評価方法は未だ確立されておらず[4]，国内外で研究が進められている．先行研究の主たる対象者は，プログラミングやオブジェクト指向設計・分析，データベース設計に関する学習を済ませた学部生や大学院生と若手技術者である[2,5]．前提知識がほとんどない対象者，例えば大学入学直後の学生を対象とした研究事例も少数ながら報告されている[6]．しかし，初学者によるモデリングを対象とした定量的な分析に基づく評価基準の提案はない．

2. 研究目的

本研究の目的は，初学者を対象とした，概念モデリングに関する評価方法を探究することにある．ここでの初学者とは，高校生や大学に入学したばかりの学生など，専門的な勉学を始める前段階にある学習者，すなわちシステム開発の手順やプログラミング等に関する前提知識を有していない学習者を対象としている．まず，これらの初学者に対して，概念モデリングとして「対象世界の構造を，ある一定の書式に従い，図として記述する事」を課した場合，どのような誤りが生じやすいのかを分析する．その上で，初学者に対して概念モデリングの基礎を確実に定着させるための支援機能として，誤り自動検出機能を開発する．

†1 信州大学大学院理工学系研究科
Graduate School of Science & Technology, Shinshu University
†2 信州大学工学部
Shinshu University, Faculty of Engineering

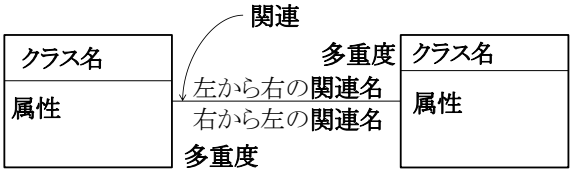

図 1. 本研究で用いるクラス図の記法

3. 初学者に対するモデリング学習

大学入学直後の工学部情報系学科の学生にモデリング学習を行い，同一課題への解答結果を比較することで，モデリング初学者の誤りの傾向を考察する．

3.1. 本研究で扱うクラス図

本研究では，対象の全体像を静的な特徴として表現するクラス図をモデリングの記法として取り上げる．クラス図の記述要素として，図 1 のようにクラス名，属性，関連，関連名，多重度のみに着目する．一般的なクラス図にみられるメソッド，属性の型などは記述対象外とした．

3.2. 被験者

被験者は，大学入学年度の違う初学者 3 グループ(以下，11T 群，12T 群，13T 群)247 名とする．被験者の違いは，表 1，表 2 のようにクラス図の導入方法が異なる．全グループにおいて，必修科目の新入生ゼミナールの一環として，モデリング教育を導入する．ここでは，オブジェクト指向の概念把握を主眼とするのではなく，UML を用いたモデルベースの思考訓練を第 1 義とする．また，11T 群および 12T 群に対しては，2 年次に応用的なモデリング科目を導入する．ここでは，システム設計の基礎を学習することが第 1 義とされる．

モデリングの際に必要な能力の内，概念形成能力を，本研究においては「記法を正しく用いて作図する能力」，要求分析能力を「課題の要求に矛盾することなく作図する能力」，そして抽象化能力を「モデル化対象に不要または不適切なクラスや属性の定義を回避する能力」とする．そして，モデル化の対象はコンピュータ上で動作可能なサービスに限定せず，日常世界に存在する事物全般とした．その理由は，プログラムや実行コードを意識せずに，対象をモデル図で表現することに注力させるためである．この教育活動における当初の教育目標は，「対象事物を，記法的な誤りなく，クラス図を用いて表現できること」とした．この目標を達成するよう，授業での説明内容が準備され，実験に先立つ講義で利用される．

3.3. 実験概要

本実験は読解・記述・修正の 3 種に分類される．以下にそれぞれの概要を説明する．

・記法を確認させる目的で，図から情報を読み取り，設問に答える読解課題．

表 1. 被験者群の特徴

	11T群	12T群	13T群
人数	86	88	73
実験年次	1年次, 2年次	1年次	1年次
前提知識 (学習期間)	アルゴリズム的思考法 (7時限)	なし	なし
実験時期	2011/6〜7月 2012/10月	2012/5〜6月 2013/10月	2013/4〜5月
授業形態	必修(2011) 選択(2012)	必修(2012) 選択(2013)	必修

表 2. 被験者群毎の授業構成の違い

段階	11T群	12T群	13T群
1	概念モデリングの定義		
	—		オブジェクト図の記法
	クラス図の記法		
	人の手を題材とした、クラス図作成演習		
	—	4種の多重度 [1, 0..1, 1..*, *]の違いの説明	
	—		クラス作成演習で生じた誤りレビュー
2	クラス図読解・記述・修正演習		
3	—	クラス読解・記述・修正演習で生じた誤りレビュー	

・与えた条件を基にクラス図記述を行う記述課題．
・誤りが含まれたクラス図から誤りを指摘し，正しいクラス図を記述させる修正課題．

これらの実験課題は全て冊子に印刷されており，解答は解答用紙へ記述させた．また，読解・修正課題においては 2 種の実験を行った．

4. 誤り分析に基づいた評価基準

読解実験の結果から，クラス図の記述要素に対する初学者における難易の程度を確認した．その結果，属性が最も難しく，次いで多重度，関連，そしてクラスという順で易しくなる傾向が見られた．この結果に基づき，記述実験の結果を整理し，誤りパタンを詳細に分析した．その結果，記述要素は 3 種(書式，クラス，関連)であり，誤りカテゴリは 4 種(記法誤り，クラス誤り，属性誤り，関連誤り)となった．4 種のカテゴリ毎の発生率を図 2 に示す．全被験者グループにおいて，属性誤りの発生率が最も高い．

以下，誤りカテゴリ毎に詳細な誤りパタンと，それぞれの発生率および誤りの具体例を示す．

4.1. 記法誤り

このカテゴリには，所定の書式に従って記述されていない解答が含まれる(記法誤り)．具体的には，本来関連が 1 つである箇所が 2 つになっているパタンや，関連 1 つにつき関連名が 1 つしか記述されてい

ないパタンが挙げられる．発生割合は，全被験者グループにおいて，全体の約 10%であった．

記法誤りの具体例を図 3 に示す. このクラス図は，洋菓子職人クラスとお菓子クラスから成り，関連が 2 本引かれている．そのため，関連名と多重度は本来 4 つずつ記述されていなければならない．しかしこの例では，関連名および多重度は 2 つずつであり，必要な要素が記述されていない.

4.2. クラス誤り

クラスに関連する誤りは，クラス誤りと属性誤りに分類される．クラス誤りは，抽象度の異なるクラスが同時に存在している(抽象度混在)誤り，同じクラス名を持つクラスが複数記述されている(同クラス)誤り，クラス名と全く同じ属性が記述されている(クラス属性同じ)誤り，クラス名の記述が無い(クラス名無し)誤りの 4 種に分類される．発生割合は，抽象度混在がおよそ 15%，同クラスがおよそ 5%，クラス属性同じが 40%，クラス名無しが 15%程度であった．

クラス誤りの具体例として，同クラスの例を図 4 に示す．3 つあるクラスの内，菓子職人クラスが 2 つ存在している．これは，同名のクラスが 1 つのモデルに複数存在する誤りである．

4.3. 属性誤り

属性誤りは，クラスに含まれる属性が不適当なものを指す. これには, クラス内に属性の記述がない(属性無し)誤り，複数のクラス内に全く同じ属性が記述されている(属性同じ)誤り，属性に具体値やクラスを構成する部品を記述している(具体値)誤り，属性にクラスの数量に関する記述がある(数量)誤り，属性にクラスを利用して実現したいメソッドが含まれる(メソッド)誤り，一つのクラス内に同義の属性が存在する(同義属性)誤り，一つのクラス内に同一名の属性が存在する(同名属性)誤り，クラスに全く関係の無い属性が存在する(クラス属性関係無し)誤りの 8 種に分類される．発生割合は，具体値が約 50%を上回り，最も多く見られた．次いで属性同じが 25%程度であり，属性無しが約 15%と続く．

属性誤りの具体例として，属性無しの例を図 5 に示す．3 つあるクラスの内，菓子職人および洋菓子職人のクラスにおいて，属性の記述が無い．これは必要な要素が記述されていない誤りである．

4.4. 関連誤り

関連誤りは，関連誤りと多重度誤りに分類される．関連誤りは，関連名の記述が無い(関連名無し)誤り，関連名として不適切な記述が含まれている(関係名誤り)誤りの 2 種に分類される．多重度誤りは，多重度の記述が無い(多重度無し)誤り，多重度の判定が誤っている(多重度誤り)誤りの 2 種に分類される．発生割合は，関連名無し・多重度無しがどの被験者グループでも高く，12T 群と 13T 群が 15%，11T 群が 5%であった．また，関連名誤りと多重度誤りは，11T 群と

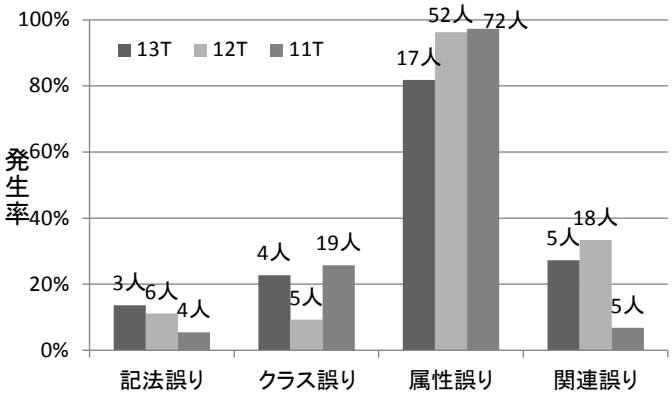

図 2. 誤りカテゴリ毎における発生率

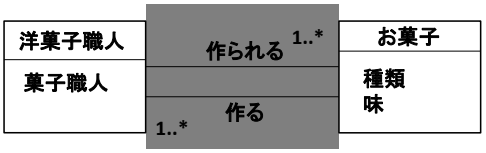

図 3. 記法誤り具体例

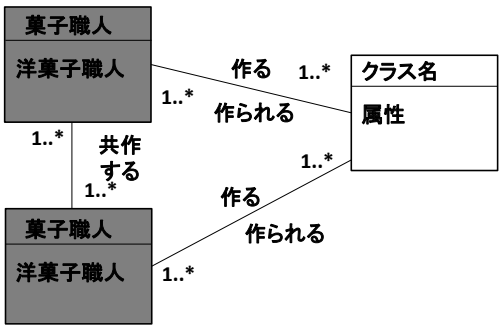

図 4. クラス誤り具体例(同クラス)

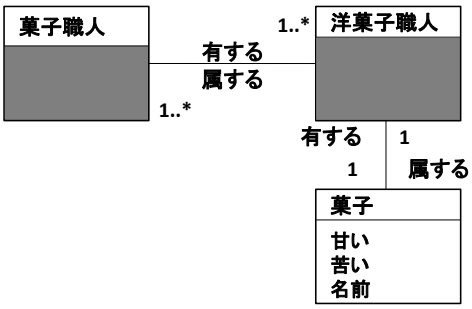

図 5. 属性誤り具体例(属性無し)

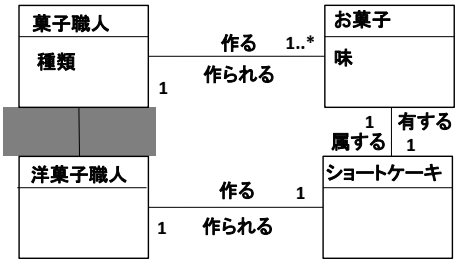

図 6. 関連誤り具体例(関連名無し･多重度無し)

表 3. 初学者のクラス図における評価基準

記述要素	カテゴリ	略称	説明
書式	記法誤り	記法誤り	書いてあるが書式が違う
クラス	クラス誤り	抽象度混在	抽象度の異なるクラスを記述する
		同クラス	全く同じクラスを複数記述する
		クラス属性同じ	クラス名と属性名が全く同じ
		クラス名無し	クラス名の記述が無い
	属性誤り	属性無し	属性が記述されていない
		属性同じ	複数のクラスに同一セットの属性を記述する
		具体値	属性に具体値やクラスを構成する部品を記述する
		数量	属性にクラスの数量に関する記述がある
		メソッド	属性にクラスを利用して実現したいメソッドが含まれる
		同義属性	一つのクラス内に名前の異なるが同じ意味の属性を記述する
		同名属性	一つのクラス内に同一名の属性を記述する
		属性関係無し	クラスに全く関係の無い属性が書かれている
関連	関連誤り	関連名無し	関連名が記述されていない
		関連名誤り	関連名が書かれているが内容に誤りがある
		多重度無し	多重度が記述されていない
		多重度誤り	多重度が書かれているが内容に誤りがある

13T 群においてほぼ見られなかった．また，12T 群においては，関連名誤りが約 10%見られた．

関連誤りの具体例として，関連名無しと多重度無しの例を図 6 に示す．洋菓子職人クラスと菓子職人クラスの関連に関連名および多重度の記述が無い．記法誤りと違う点は，1 本の関連に対して関連名および多重度が一切記述されていない事にある．これは必要な要素が記述されていない誤りである．

4.5. 評価基準

前節までの結果をふまえ，初学者のクラス図に対する評価基準として 17 項目に分類した(表 3).

これらの項目により記述課題の解答に含まれる誤りをもれなく分類できた．また，修正実験に対しては，修正課題を組み立てる根拠とし，さらに回答を整理する基準ともなった．また，本評価基準は，前提知識やモデリング学習の方法が異なる 11T 群，12T 群，13T 群のいずれの解答における誤りを分類するのに利用することができた．これらのことから，情報システム設計・開発に関する前提知識を有しない初学者による概念モデリングを評価する基準として，提案基準は妥当ではないかと考える．

5. 誤り自動検出機能の開発

前章で示した結果は，クラス図を机上で記述させる形式でのモデリング学習を対象としていた．そのため，解析は紙に記述されたモデル図に対して行ってきた．この解析の自動化を図るべく，ソフトウェア開発設計支援ツール astah*[7]でのモデリングを対象にした自動検出機能の開発を行う．ここでの自動検出機能とは，評価基準に基いて astah*で記述されたクラス図から誤りパタンを自動検出する機能である．

5.1. 評価基準の再分類

表 4. 再分類された評価基準

	記述要素	カテゴリ	略称
要求非依存 / 全自動評価	書式	記法誤り	記法誤り (※)
	クラス	クラス誤り	同クラス (※)
			クラス属性同じ
			クラス名無し
		属性誤り	属性無し
			属性同じ
			同名属性 (※)
	関連	関連誤り	関連名無し
			多重度無し
要求非依存 / 半自動評価	クラス	クラス誤り	抽象度混在
		属性誤り	メソッド
			同義属性
要求依存 / 半自動評価	クラス	属性誤り	具体値
			数量
			属性関係無し
	関連	関連誤り	関連名誤り
			多重度誤り

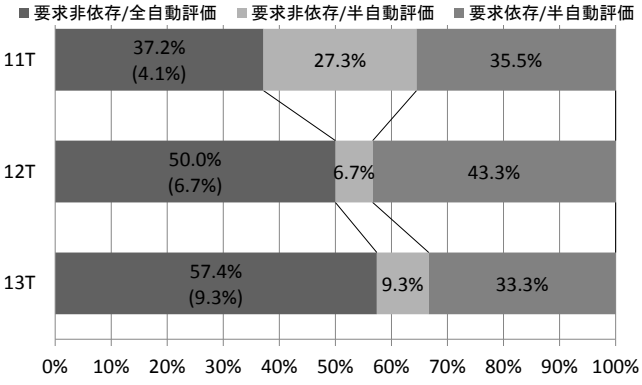

図 7. 再分類された評価基準毎の発生率

自動検出機能の開発に伴い，表 3 で示した評価基準を再分類した(表 4)．再分類するに当り，全自動評価の可否，要求依存による有無に着目した．

新たな分類は，要求非依存かつ全自動評価可能な

項目，要求非依存かつ半自動評価可能な項目，要求依存かつ半自動評価可能な項目の 3 種である．この 3 種の評価分類毎の発生率を図 7 に示す．どの被験者グループにおいても要求非依存かつ全自動評価が可能な誤りの発生率が平均で 48.2%と最も多く，次いで要求依存かつ半自動評価可能な誤りが 37.4%，要求非依存かつ半自動評価可能な誤りが 14.4%の順で発生率が高い．

本研究では，どのグループでも発生率の高い，全自動評価が可能な 9 項目の誤りを自動検出機能で検出する事とした．これらの誤り検出のために，astah*へ組込む plugin として自動検出機能を開発する．これら 9 項目の内，記法誤り・同クラス・同名属性の 3 項目は astah*の仕様で発生しない(表 4 内※)．よって残り 6 項目の誤りを自動検出する機能の開発を行う．発生しない 3 項目の発生率を図 7 の括弧内に記載した．この時の発生率は，要求非依存かつ全自動評価可能な項目内での発生率ではなく，全体での発生率である．astah*で発生しない 3 項目(平均で 6.7%)に比べ，6 項目(41.5%)の発生率が高い事がわかる

これらの 6 項目は更に 2 つのグループに分類できる．それは，クラス名無し・属性無し・関連名無し・多重度無しの「要素が足りない項目」と，クラス属性同じ・属性同じの「同じ要素が存在する項目」である．「要素が足りない項目」に関しては，astah*ではクラスを記述する際，デフォルトで名称が記述されるため，クラス名が取り出せない事がない．一方，属性無し・関連名無し・多重度無しはデフォルト値がないため，取り出せるかが判断の基準となる．「同じ要素が存在する項目」に分類される 2 種類の誤りパタンの違いは，クラス単位の誤りかどうかである．クラス属性同じでは単一クラス内での比較に対し，属性同じでは複数のクラスを比較し，更には取り出したクラス名と属性を記憶する必要がある．

5.2. astah* plugin の開発

astah*へ自動検出機能を組込む事で，モデル図を記述する際に誤りがすぐに指摘され，修正が容易になる．また，提出された astah*ファイルに記述されたクラス図からは，全自動評価が可能な誤りパタンの発生が無くなる．自動検出機能を astah*へ組込む方法として astah*の plugin として実装する事が考えられる．astah*の plugin 機構は，astah*の起動時にインストールされたフォルダ配下の plugins フォルダ内にある plugin ファイルをロードする．学習者には予め自動検出 plugin ファイルを保存しておくだけで本機能を使えるようになる．

astah* plugin としての実装項目を次に示す．

1. 拡張ビューおよび新規タブの作成．

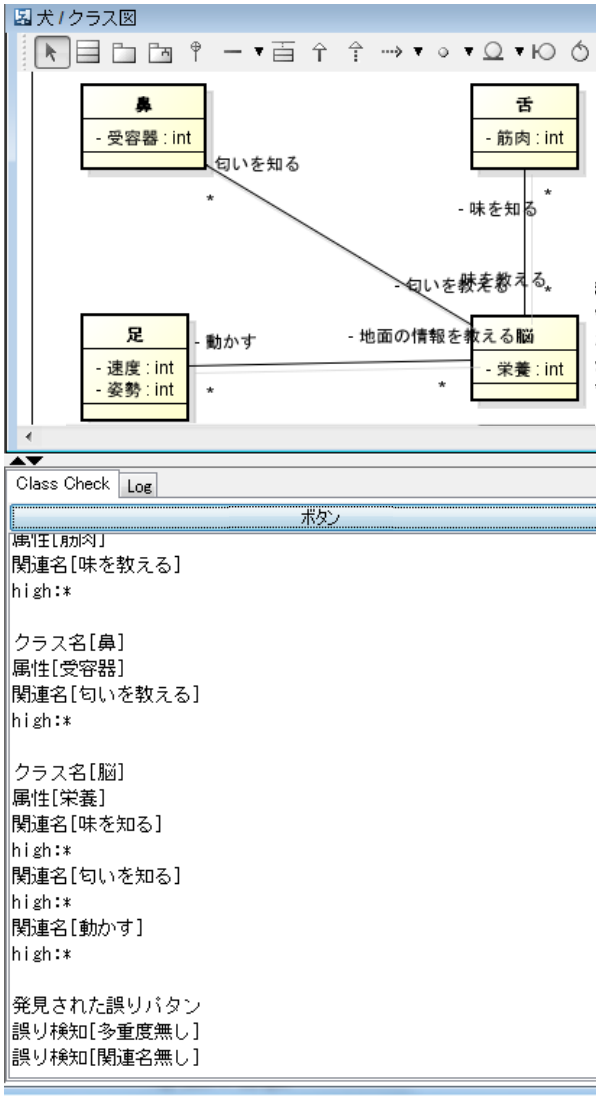

図 8. astah*画面

2. 自動検出結果を新規タブへ出力．
3. ボタンによる誤り検出の出力制御．
4. 出力結果を過去ログとして保存．
5. 出力回数のカウント．

まず，astah*上に出力するための新規タブ作成を行う．新規タブは astah*の拡張ビューに独自タブとして追加する．

次に，自動検出機能の出力結果を作成した新規タブへ出力する．出力する情報は，クラス図の持つ要素(クラス名・属性・関連名・多重度の上限・下限)，発見された誤りパタンである．これらの出力結果をタブへ表示するために，IPluginActionDelegate インタフェースを用いる．

ボタンによる出力の制御は，出力結果を過去ログに保存するために必要となるアクションである．ボタンが押される事で誤り自動検出が機能し，現在記述されているクラス図が評価される．また，出力結

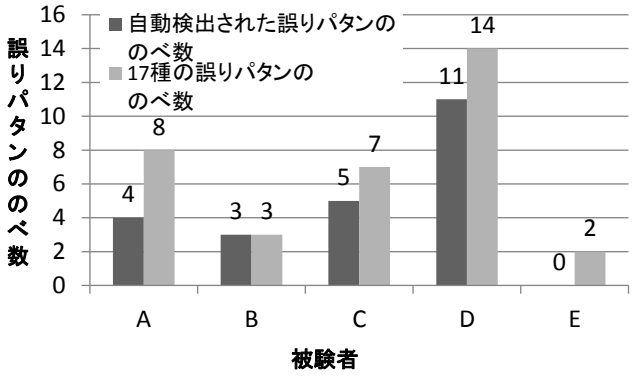

図 9. 評価実験における誤りパタンの発生数

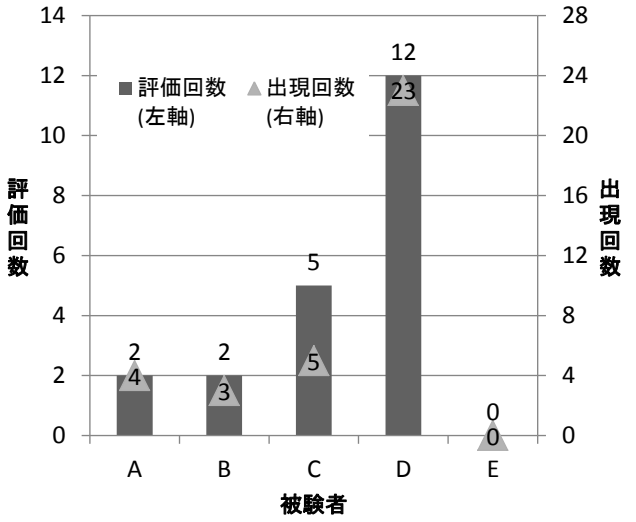

図 10. 評価実験におけるエラーの評価回数と
エラーの出現回数

果を過去ログに保存することで，出力回数毎の誤りパタンの推移が記録される．

実際に astah* plugin を導入し，記述されたクラス図の誤りを自動検出している様子を図 8 に示す．学習者は，図中下部に示された誤りの検出情報に基づき，自身のモデル図を修正していくことになる．

6. 誤り自動検出機能の評価

5.2 で示した誤り自動検出機能の有効性を評価した．

6.1. 実験概要

被験者は情報工学専攻の大学院生 5 名(以下，A, B, C，D，E)とした．この 5 名は，モデリング学習は未経験であるが，プログラミング言語やアルゴリズム的思考法等，情報工学の基本的な知識を有している．被験者にクラス図の説明および astah*の使い方を説明した上で，クラス図の多重度に関する課題を出題し，astah*を用いてクラス図を記述させた．この時，astah*plugin は未使用である．次に，plugin を適用し，記述されたクラス図の修正を行わせる．この時に，誤り自動検出機能の有するボタンのカウント数，発見された誤りパタンの内容，誤りパタンの出現数をチェックする．

課した問題は全部で 5 問である．問 1・問 2・問 3 はそれぞれ問われる多重度が異なる．問 4 は関連に着目した課題である．問 5 は犬をモデル対象とし，構成するパーツをクラス名として表現できるか，パーツ間の関連および多重度の関係を正しく記述できるかが問われる．

6.2. 実験結果

誤り自動検出機能を使用した後のクラス図からは，どの被験者も誤りが見られなかった．評価実験における誤りパタンの発生数を図 9 に示す．ここでは，今回開発した検出機能で同定される 6 種の誤りパタンののべ発生数(左の棒)と，半自動評価項目を含む全ての誤りパタンののべ発生数(右の棒)を示す．5 名の被験者の平均で 7 件の誤りが生じていた．また，どの被験者も全誤りパタンの半数以上が全自動評価可能な項目であった．全被験者において発生した誤りパタンの内容は，関連名無しが最も多く 9 件であった．次いで多重度無しが 4 件，同じ属性が 3 件見られた．クラス名無し・クラス属性同じ・属性無しについては 1 件も発生していない．全自動評価可能な項目以外では，多重度誤り 5 件，関連名誤り 2 件，抽象度混在 2 件，具体値 1 件，数量 1 件が見られた．

また，誤りを含まないクラス図に到達するまでに要したボタンのカウント数および誤りパタンの出現数を図 10 に示す．被験者が誤り自動検出機能を使用するためにボタンを最低 1 回はクリックしているため，ここでのカウントは実際のクリック数から 1 回分減らした数値となっている．5 名の被験者の平均で，評価回数は 4 件，エラーの出現回数は 7 件であり，評価 1 回当たりのエラーの出現回数は約 1.7 件である．

6.3. 考察

この評価実験の結果から，誤りパタンが発生したクラス図を誤り自動検出機能の使用で誤りの軽減が確認できた．よって，初学者におけるクラス図の評価基準の内，全自動評価が可能な 6 項目は，本機能を用いる事で出現を抑制できると考えられる．

また，問 5 では誤りパタンの出現が 5 人中 3 人と，他の問に対して多めであった．特に「同じ属性」パタンの出現が最も多い．他の問ではクラスを 2 つ記述するが，問 5 は最低でもクラスを 4 つ記述しなければならない．そのため，モデル図として複雑になり，関連および属性の確認で見落としが生じることが考えられる．

評価回数が 12 回と極めて多い被験者 D は，クラス図上に誤って記述した関連を非表示にしていただけ

で，モデル図から削除していなかった．この場合，一見，クラス図に誤りが無いように見えるが，クラス図内部の情報を確認すると，関連名および多重度の記述のない関連が存在する．

7. おわりに

本研究では，クラス図を用いた基礎的な概念モデリングにおける誤り分析を行い，それに基づき初学者向け誤り自動検出機能の開発を行った．まず，大学入学直後の初学者 3 グループを対象に机上でクラス図のモデリング学習を行い，読解・記述・修正の 3 種類の実験課題を課した．実験結果の解析から，初学者のクラス図における誤りパタンとして 17 項目の評価基準としてまとめた．

この結果に基づき，ソフトウェア開発設計支援ツール astah*でのモデリングを対象にした誤り自動検出機能の開発を行った．その際，誤りパタンを全自動評価の可否，要求依存による有無に着目して再分類した．本研究では，全自動評価が可能であり，要求に依存されない 9 項目の誤りの内， 6 項目の検出が可能なプログラムを開発した．残りの 3 項目は，astah*の仕様で発生しないこととなる．その後，astah* plugin 化を行った．

開発した astah* plugin の有効性を評価するため，5 名の被験者を対象にクラス図を記述する課題を astah*で課した．評価実験により，誤りパタンが発生したクラス図を誤り自動検出機能の使用で誤りの軽減ができ，初学者におけるクラス図の評価基準の内，全自動評価が可能な 6 項目の発生を抑制ができた．

今後は本研究で開発した誤り自動検出機能の改良が求められる．具体的には，誤り検出結果の出力をよりわかりやすい表記にする等が挙げられる．また，初学者によるモデリングに対する検出機能の有効性を評価するため，大学入学直後の学生を対象にした授業への導入を検討する．そして，初学者が記述したクラス図において，今回検出対象とした誤りの発生を軽減する事を確認できるか評価実験を行う．また，本機能では，初学者におけるクラス図の評価基準 17 項目の内，9 項目の実装を行った．残りの 8 項目，すなわち全自動化が不可能かつ要求に依存されない 3 項目および全自動化が不可能かつ要求に依存される 5 項目の実装を検討する．

謝辞 本研究は科研費 22300286 の助成を受けた．

参考文献

1) 児玉公信：“情報システム設計における概念モデリング”，人工知能学会誌,Vol. 25, No.1, pp.139-146, 2010.
2) IPA：“モデルベース設計検証技術者スキル体系化調査調査報告書”, 2012.
3) S. Sendall : “Model Transformation : The Heart and Soul of Model Driven Software Development”, IEEE SOFTWARE , Vol.20, No.5, pp.42-45, 2003.
4) J. Bezivintal et.al., : “Teaching Modeling : Why, When, What?”, pp.55-62, MODELS 2009, 2009.
5) 中尾信明：“オブジェクト指向，UML に関する教育の視点と分析”，情処研報, Vol.2004-CE-74, No.2, pp.9-16, 2004.
6) 長尾祐樹他：“初心者用 UML の提案とその評価”，情処研報, Vol.2008-CE-97, No.7, pp.45-52, 2008.
7) astah*, http://astah.change-vision.com/ja/, (2015/2/13 accessed)