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

2009－SE－163 (19)
2009／3／18

演習教育を対象としたUML設計の詳細度判定手法の提案

川崎 結花†, 上田 賀一†
† 茨城大学

オブジェクト指向設計において，初心者が詳細設計を行う場合，詳細さの統一がとれていない設計仕様を作成してしまう場合がある．本研究では，オブジェクト指向ソフトウェア設計演習を対象に，設計詳細レベルを段階的に達成していくことを支援する詳細度判定手法および設計の品質評価手法を提案する．あらかじめ各ステップにおいて指導者が詳細さを図要素によって定義し，指導者は学習者によって作成されたUML設計図の詳細度および設計品質を測定し，測定結果をもとに学習者が設計を修正する作業を繰り返して設計の詳細さを高めていくプロセスを考案した．詳細度を測定するために，設計の詳細さを表す詳細度を定義し，メトリクスを用いた詳細度の算出法を検討した．

A Method of Judging Refinement Level of UML Design Model for Practical Education

Yuika KAWASAKI† and Yoshikazu UEDA†
†Ibaraki University

When beginners design software in object-oriented technique, they often make the design specifications that detailedness is not uniform. We propose a judgment method of details level to achieve design details level step by step for practical education. The process we devised is as follows: (1) teacher defines refinement by diagram elements for each step, (2) teacher measures the refinement level of UML diagram that learners made in every step, (3) learners raise the refinement level of their design by repeating to revise a design with measurement results. We defined a refinement level to express the detailedness of the design and examined the judgement method of the refinement level by metrics.

1 はじめに

近年，大学等の教育機関において，オブジェクト指向によるソフトウェア設計やUMLの記法をカリキュラムに取り入れるケースが増加している．しかし，オブジェクト指向設計には明確な答えが無く，学習者個々の設計を見て指導を行う必要があり，指導者の負担が大きい．そのため，学習者が自分で学習することを可能にするための手法が提案されている．具体的には，UMLの記法の正しさ等を自動で評価するシステム [4] や，UML図から設計品質を評価するモデル [2][3] 等である．

上記のような様々な手法が提案されているが，その他に詳細なUML設計図を記述することを支援する手法が必要とされる．設計は詳細設計工程の進行とともに，段階的に図要素が追加され詳細さが高められていくべきである．しかし，初心者はどのように，どの程度まで図要素を追加すべきか分からない場合が多く，詳細な設計を行わせるためには，設計の進行に応じて，追加すべき図要素を学習者にその都度示していくことは有効である．

そこで本研究では，複数のメトリクスを用いて，オブジェクト指向設計演習において設計詳細レベルを段階的に高めていくことを支援する手法を提案する．メトリクスとはソフトウェア開発の際の成果物の様々な特性を定量的に計測し，数値化するための数学的尺度であり，ソフトウェアのフォールト予測 [1] 等に利用されている．メトリクスを用いて要素

の有無を判断して図の詳細さを評価し，不足要素を示すことができるようになれば，先に述べた初心者による詳細設計の学習を支援する有効な手法になると考えられる．

本報告では，提案手法である詳細度の測定方法について述べ，本手法の評価のための例題，使用するメトリクスや詳細度の設定について説明したのち，本手法の適用の過程を示し有効性の検討を行う．最後に本研究のまとめと今後の課題について述べる．

2 提案手法

2.1 概要

本手法は，ソフトウェアの設計工程を対象としている．設計工程を複数に分け，分割した各工程で以下の手順が1度ずつ行われる．また，評価の対象とする図をクラス図とシーケンス図とし，開発言語を規定しないソフトウェア設計とした．

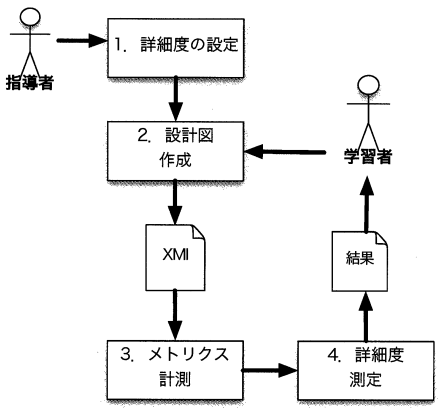

図 1: 提案手法の概要

本手法の流れは以下のようになる．また，概要を図1に示す．

1. 指導者が分割した工程において記述されるべき図の要素を指定する
2. 学習者はそれを参考にCASEツールでUML設計を行う
3. CASEツールが出力するXMIからメトリクスの計測を行う
4. メトリクス値から，詳細度を測定する
5. 学習者は結果から設計図を修正する
6. 詳細度が規定以上になるまで2〜5を繰り返す

2.2 詳細度判定

以下で，詳細度判定の方法の詳細について述べる．

2.2.1 詳細度の定義

詳細度はレベルで表す．今回は一例として，レベルを1，2，3の3段階とし，レベル2がその工程で必須の図要素が記述されている状態，レベル3がレベル2の条件を満たしさらに詳細な図要素が記述されている状態を表すことにした．指導者がレベル2，3に当たる図要素を指定することで，レベル達成の条件を指導者が意図したものに設定することができる．

表 1: 詳細度

レベル	内容
レベル1	n番目のステップでの初期値
レベル2	n番目の設計ステップで必須の要素が記述されている
レベル3	n番目の設計ステップでレベル2の条件を満たしさらに詳細な図要素が記述されている

設計工程を図2のように4〜5程度の小さなステップに区切り，1ステップごとにレベル達成の条件を設定する．例えば，1ステップ目にはクラスとメソッド，属性がクラス図に記述されていればレベル2，属性の型まで記述されていればレベル3とし，2ステップ目ではレベル2の条件としてクラス，メソッド，属性，属性の型が記述されていること，レベル3の条件としてメソッドの引数が記述されていること等というように定めることができる．

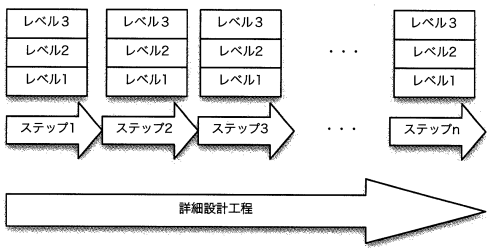

図 2: ステップとレベル

学習者は各ステップにおいてレベル2以上の詳細度に達成しなければ次のステップに進むことができない．このようにステップごとにレベルを達成していくことで，工程の最後には指導者が意図した詳細度の設計図が作成できると考えられる．

2.2.2 詳細度の測定

詳細度は，指定した図の要素が記述されているかどうかをメトリクス値から判断し，詳細度を決定する．以下にその詳細を示す．

図要素の記述判定 一つの図要素につき一つのメトリクス値，または複数のメトリクス値を計算式に当てはめて算出した値のどちらかを使用する．図に要素が記述されているか否かは閾値によって判断する．閾値は各要素に対し一つ設定する．

例えば，クラスにメソッドが記述されているか否かは，メトリクス NOM によって判断し，NOM の値が 0 なら記述されていない，1 以上なら記述されていると定める．なお，その他の例は 3.2 節で説明する．

詳細度の決定 詳細度はクラス毎と設計図全体という 2 種類の詳細度を測定する．クラス毎の詳細度の決定は，レベル 2 で指定した全ての要素が記述されていたらレベル 2 とし，記述されていない場合はレベル 1 とする．あるクラスがレベル 2 の条件を満たし，かつレベル 3 で指定した要素が全て記述されていればレベル 3 とする．

全体の詳細度はクラスのレベルの平均で決定する．クラス毎と全体の 2 種類の詳細度を比較することで，図の中のクラスの詳細さのばらつきをとらえることができる．例えば，10 個のクラスのうち 8 個がレベル 1，2 個がレベル 3 の詳細度だったとすると，全体の詳細度は 1.4，四捨五入して 1 となり，2 つのクラスは詳細に記述できているが詳細でないクラスが多くあり，記述の詳細さに差があることが分かる．

また，詳細度の判定法の例外としてシーケンス図がない場合はその他の図要素が記述されているか否かに関わらず，詳細度はレベル 1 とする．

3 適用例

この章では提案手法の有効性を確認するためにケーススタディへの適用を行う．

3.1 実験に用いたメトリクス

実験には佐藤らの品質評価モデル [2][3] で使用している品質評価および欠陥検出に用いられているメトリクスを使用した．表 2 に使用するメトリクスとその説明を示す．

表 2: 詳細度判定で使用するメトリクス

メトリクス	概要
NOA	クラスの属性数
NOM	クラスのメソッド数
NPM	クラスの private メソッド数
NOP	メソッドの引数の数
RAC	クラス総数における抽象クラス数の割合
NPAM	クラスの Public アクセサメソッドの数
DIT	継承木の深さ
RMNP	クラスのメソッド総数における 引数を持たないメソッド数の割合
NMCCC	集約元クラスから集約先オブジェクトへの メッセージの数
NSV	あるクラスのクラス変数の数

3.2 詳細度の定義

詳細度は表 5 に示す通りに設定を行った．これは「ゼロから学ぶソフトウェア開発完全入門」[6] の第 5 部を参考にしたクラス設計の手法で，最初に必要なクラス等を洗い出し，細部を記述した後，継承等を用い再利用性などを高める処理を行うことを想定している．

また，各要素に対応するメトリクス値を表 3 に示した通りに定めた．表 3 において，属性の測定範囲が全体となっているのは，3.3 節で取り上げる例題に，属性を持たないクラスがあるため，属性の記述をクラス共通の条件として定めることができなかったためである．また，メソッドの可視性の判定で，private メソッドが無い場合は記述なしと見なす判定条件を設定している．実験で使用する CASE ツールの仕様上，メソッドを追加すると自動で可視性が public に設定されてしまい，学習者が可視性を意識しなくても記述されてしまうため，可視性が private に設定されているメソッドがあれば学習者が可視性を意識して記述した可能性が高いとして．private メソッドの有無によって記述の判断をすることにした．

3.3 適用実験

3.1，3.2 節で設定した条件に基づいて，例題に本手法を適用し，本手法が有効であるかどうかを確認する．例題には書籍「初めて学ぶ UML」[5] 8 章のケーススタディを用いた．

例題はコンビニエンスストアシステムの販売処理と商品追加処理を実現するシステムの設計であり，ステップ 1 の時点でクラス数が 11 個，そのうち 2

表 3: 要素に対応するメトリクス値

要素	使用するメトリクス値	測定範囲	閾値 記述なし
メソッド	NOM+NPAM	クラス	0
属性	NOA	全体	0
引数	RMNP	クラス	1
クラス変数	NSV の全クラスの平均	全体	0
メソッドの 可視性	(NOM-NPM)/NOM	クラス	1
アクセサ メソッド	(NOA-NPV)×2-NPAM	属性を持 つクラス	0 以外
集約	NMCCC	全体	null (測定不可)
継承	DIT	全体	1
抽象クラス	RAC	全体	0

組が集約関係を持っている．本研究ではこの例題に変更を加えて詳細設計を作成していった．図 3〜5 に実際に設計したクラス図，表 4 にステップ毎の設計の変更点を示す．

また，設計図の作成には CASE ツールの IBM Rational Rose XDE を使用し，出力した XMI を使用してメトリクスを計測した．

表 4: ステップ進行による変更

ステップ	変更部分
1	・可視性を考慮せずに属性を追加 ・可視性を考慮せずに引数なしメソッドを追加
2	・属性を持つクラスにアクセサメソッドを追加 ・2 つのクラスに private メソッドを 1 つ追加 ・メソッドに引数を追加
3	・2 つのクラスに継承関係を追加し， スーパークラスを一つ追加
4	・スーパークラスを抽象クラスに変更
5	・2 つのクラスにクラス変数を 1 つ追加

3.3.1 詳細度

表 6〜8 に，ケース 1 のステップ 2〜4 の完了時点でのクラスおよび全体の要素の記述の有無と，判定された詳細度を示す．なお，表中の「○」はそのクラスに記述がある，「×」は記述が無い，「-」は測定対象ではないことを表す．これらは 3.1 節で述べた閾値に則って判定している．また，太い罫線から左側はレベル 2 達成の条件になっている図要素，右側がレベル 3 達成の図要素の項目である．

ステップ 2 において，実際の設計図である図 3 では全てのクラスにメソッドが記述され，測定結果である表 6 のメソッドの項目の結果と一致しており，記述の有無が正しく判定できていることが分かる．また，設計に使用した Rational Rose XDE の仕様上，設計図に引数が表示されていないが，変更箇所をまとめたの表 4 のステップ 2 の項目にある通り，すべてのメソッドに引数が記述されており，結果と一致している．属性に関しても，属性が記述されているクラスがあるので，正しい結果が得られている．しかし，設計図でメソッドの可視性が記述されているにもかかわらず，表 6 の判定結果では，多くのクラスで記述が無いという結果になっている．その他のステップも同様に，ほぼ全ての指定した UML 図の要素の有無が判定できているが，メソッドの可視性だけは，記述されているにもかかわらず記述無しと判定されているクラスが多い．このような判定結果が得られたのは，可視性を判定するメトリクスを設定する際に，3.2 節で述べた理由により private メソッドの有無によって記述の判断をすることにしており，規模が小さい例題では private メソッドを用いる必要のあるクラスが少なく，多くのクラスで可視性が記述されていないと判定されてしまったためであると考えられる．そのため，可視性の有無を判定するメトリクスや測定範囲を再検討する必要があ

表 5: 各ステップにおけるレベルの設定

ステップ	ステップ 1	ステップ 2	ステップ 3	ステップ 4
レベル 2	属性 メソッド	属性 メソッド アクセサメソッド 引数	属性 メソッド アクセサメソッド 引数 継承 集約	属性 メソッド アクセサメソッド 引数 継承 集約 抽象クラス
レベル 3	メソッドの可視性	メソッドの可視性	メソッドの可視性 抽象クラス	メソッドの可視性 クラス変数

表 6: 詳細度 (ステップ 2)

クラス	メソッド	引数	アクセサメソッド	属性	メソッドの可視性	レベル
Goods	○	○	○	-	×	2
GoodsKind	○	○	○	-	×	2
GoodsKindForm	○	○	-	-	○	3
GoodsKindMGR	○	○	-	-	×	2
GoodsList	○	○	-	-	×	2
PersisitentGoodsKind	○	○	-	-	×	2
Receipt	○	○	-	-	×	2
Sales	○	○	○	-	×	2
SalesForm	○	○	-	-	○	3
SalesMGR	○	○	-	-	×	2
TransactionManager	○	○	-	-	×	2
全体	-	-	-	○	-	2

表 7: 詳細度 (ステップ 3)

クラス	メソッド	引数	アクセサメソッド	属性	継承	集約	メソッドの可視性	抽象クラス	レベル
Goods	○	○	○	-	-	-	×	-	2
GoodsKind	○	○	○	-	-	-	×	-	2
GoodsKindForm	○	○	-	-	-	-	○	-	2
GoodsKindMGR	○	○	-	-	-	-	×	-	2
GoodsList	○	○	-	-	-	-	×	-	2
PersisitentGoodsKind	○	○	-	-	-	-	×	-	2
Receipt	○	○	-	-	-	-	×	-	2
Sales	○	○	○	-	-	-	×	-	2
SalesForm	○	○	-	-	-	-	○	-	2
SalesMGR	○	○	-	-	-	-	×	-	2
TransactionManager	○	○	-	-	-	-	×	-	2
全体	-	-	-	○	○	○	-	×	2

表 8: 詳細度 (ステップ 4)

クラス	メソッド	引数	アクセサメソッド	属性	継承	集約	抽象クラス	メソッドの可視性	クラス変数	レベル
Goods	○	○	○	-	-	-	-	×	-	2
GoodsKind	○	○	○	-	-	-	-	×	-	2
GoodsKindForm	○	○	-	-	-	-	-	○	-	3
GoodsKindMGR	○	○	-	-	-	-	-	×	-	2
GoodsList	○	○	-	-	-	-	-	×	-	2
PersisitentGoodsKind	○	○	-	-	-	-	-	×	-	2
Receipt	○	○	-	-	-	-	-	×	-	2
Sales	○	○	○	-	-	-	-	×	-	2
SalesForm	○	○	-	-	-	-	-	○	-	3
SalesMGR	○	○	-	-	-	-	-	×	-	2
TransactionManager	○	○	-	-	-	-	-	×	-	2
Manager	○	○	-	-	-	-	-	×	-	2
全体	-	-	-	○	○	○	○	-	○	2

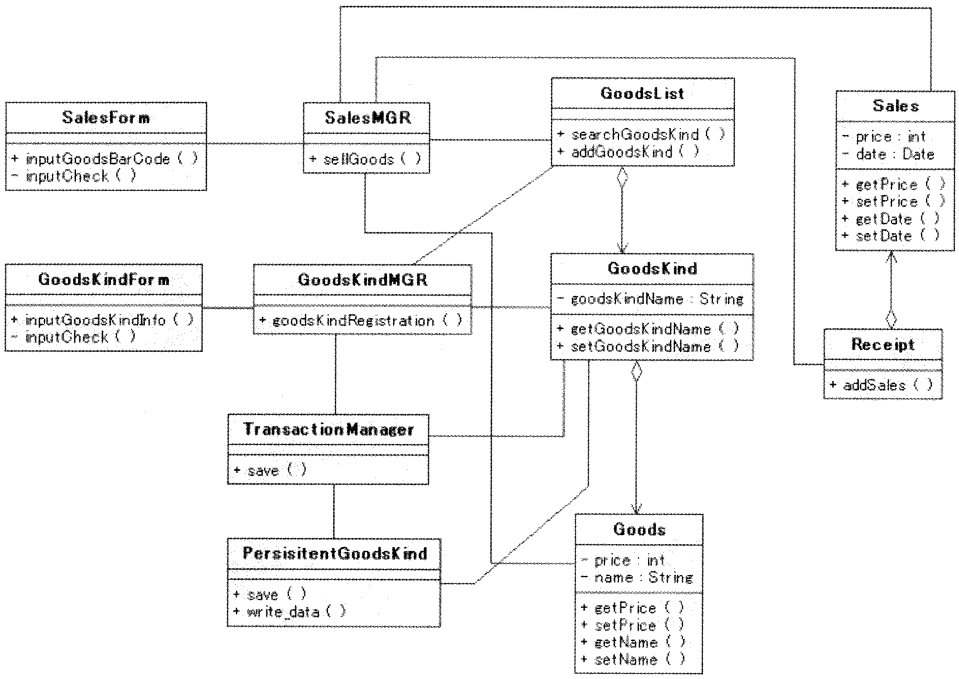

図 3: ステップ 2：クラス図

ると考えられる．

また，本実験において各図要素の記述判定に使用するメトリクス値を設定したが，これは佐藤らの品質評価モデル [2][3] で用いたメトリクスを流用したもので，必ずしもすべての図要素に対し適切に設定できているとは言えない．例えば，アクセサメソッドの記述の有無の判定に public アクセサメソッドの数である NPAM を用いているが，もしアクセサメソッドが private で記述されていた場合，図要素が記述されていると判定できない．このため，詳細度判定のためのメトリクスを新たに定義し，それらを用いて測定を行う方が適切であると考えられる．

4 関連研究

4.1 UML クラス図の評価システム

UML Class Diagram Assessor(UCDA) という CASE ツールによって作成したクラス図を評価するシステムの提案 [4] がある．この提案は指導者があらかじめ作成した模範解答と比較することで，学習者のクラス図の表記法や属性やメソッド名の命名の正しさなどを検査している．本手法ではクラス図やシーケンス図のメトリクス値のみで評価するので命名の正しさ等は検査できないが，学習者の作成する図が多様な形になっていても設計の支援をすることができる．

4.2 オブジェクト指向，UML に関する教育の視点と実践

オブジェクト指向や UML の効果的な教育の視点と実践方法についての報告 [7] がある．この提案には企業教育の経験に基づき，オブジェクト指向や UML への習得のために理解すべきポイントや教育法について述べられている．教育を対象としている本手法にも，このような方法論を取り入れることができれば，さらに効果的な教育支援が可能になると考えられる．

4.3 UML 設計の品質評価モデル

UML の設計図からソフトウェアの設計品質を評価するモデル [2][3] がある．この研究はソフトウェアの品質に関する規格である ISO9126 を参考に考案した品質評価モデルの検討を行っている．本研究の詳細度測定にこのような品質評価の手法を取り入

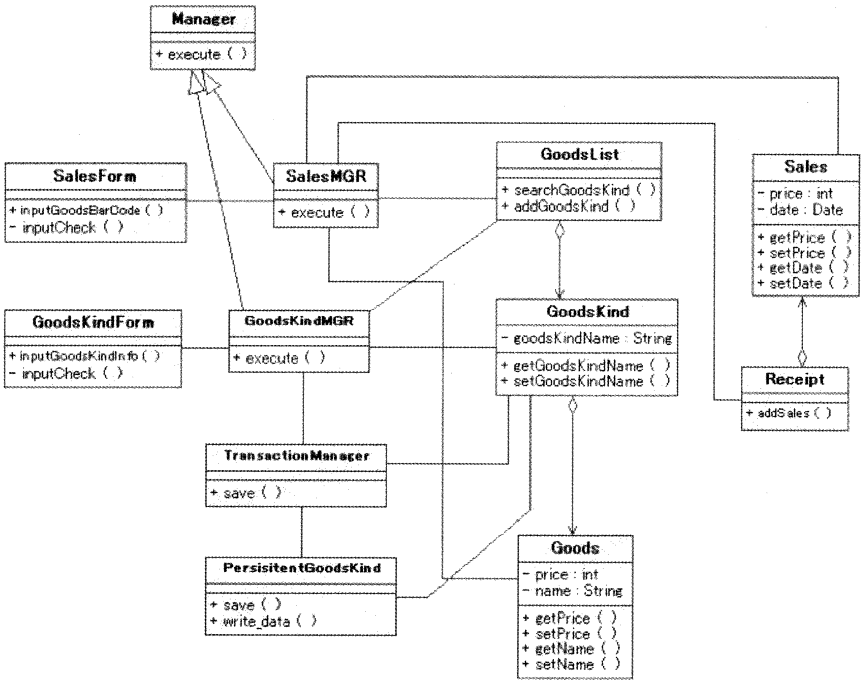

図 4: ステップ 3：クラス図

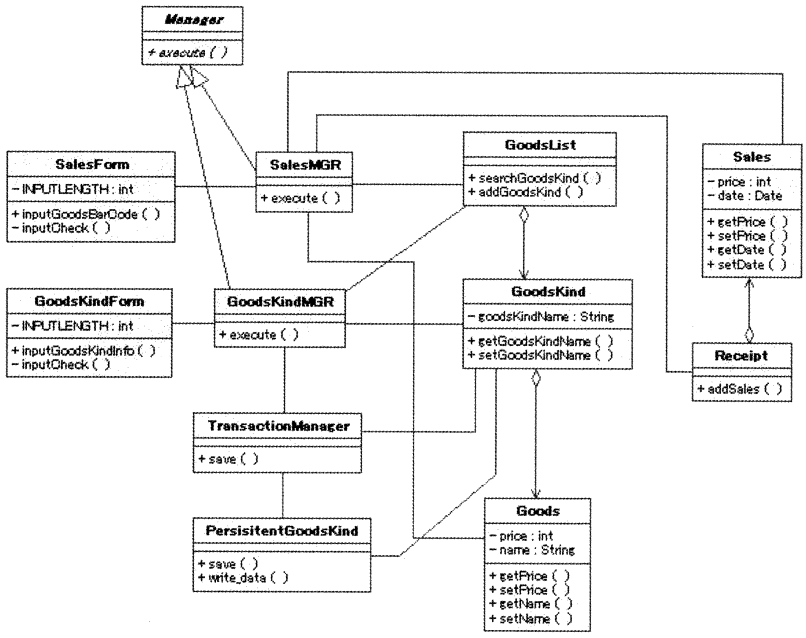

図 5: ステップ 4：クラス図

れることで，詳細設計図作成の支援だけでなく品質の高い設計の作成支援も行えるようになると考えられる．

5 終わりに

5.1 まとめ

本研究では，オブジェクト指向ソフトウェア設計の習得および詳細な設計図の作成の支援を目的に，メトリクスを利用して，設計詳細レベルを段階的に達成していくための判定手法を提案した．

設計工程を複数のステップに分割し，各ステップに指導者が達成すべき設計図の詳細さを図要素によって定義し，ステップごとに学習者が作成したUML設計図の詳細度を測定し，測定結果を元に学習者が設計を修正する作業を繰り返して設計の詳細さを高めていくプロセスを考案した．

本手法の有効性を確かめるために，専門書籍の練習問題に本手法を適用し，詳細度を測定しながら設計演習を行った．その結果，本手法により段階的に詳細度が高められていくことを確認した．

5.2 今後の課題

実例への適用 本手法は設計演習向けであるので，実際の教育現場への適用が必要である．そのためには，提案手法の手順を一貫して行うアプリケーションを作成し，演習授業にて適用する必要がある．

詳細度判定用メトリクスの定義 本研究では図要素の記述判定に品質評価用のメトリクスを用いたため，適切に判定できていない場合があった．このため，図要素の判定のためのメトリクスを新たに定義することで適切な詳細度を測定することができると考えられる．

図要素の再検討 本手法では詳細度の指定にUMLの図要素を用いたが，デザインパターンやデータクラスの使用といったより抽象的な要素を指定できればさらに有効な手法になると考えられる．そのために，例えばデータクラスを必須要素に指定し，データクラス自体の必須要素として属性とアクセサメソッドを指定するなど，階層的に要素を指定することで，より柔軟な詳細度を測定できると考えられる．

参考文献

[1] Yue Jiang, Bojan Cukic, Tim Menzies, Nick Bartlow：Comparing Design and Code Metrics for Software Quality Prediction, International Conference on Software Engineering archive Proceedings of the 4th international workshop on Predictor models in software engineering, SESSION: Defect prediction table of contents, pp.11-18 , 2008.

[2] 佐藤美穂，田村真吾，上田賀一：UML設計を対象とした品質評価モデルの検討 , 特集：ソフトウェア工学の効果と価値 分析・設計技術，情報処理学会論文誌， Vol.49， No.7，pp.2319-2327, 2008.

[3] 佐藤美穂：UML設計の品質評価モデルおよび欠陥検出モデルの検討 , 茨城大学大学院理工学研究科情報工学専攻 修士論文 2009.

[4] Noraida Haji Ali, Zarina Shukur, Sufian Idris: A Design of an Assessment System for UML Class Diagram, The 2007 International Conference on Computational Science and its Applications, pp. 539-544, 2007.

[5] 竹政昭利 著：初めて学ぶUML，株式会社ナツメ社，2003.

[6] 日経ソフトウェア 編：ゼロから学ぶソフトウェア開発完全入門 オブジェクト指向からデータベースまで，日経BP社，2005.

[7] 中尾 信明：オブジェクト指向，UMLに関する教育の視点と実践，情報処理学会研究報告，コンピュータと教育研究会報告，Vol.2004，No.49，pp.9-16，2004.