設計段階における抽象度を考慮したソフトウェアの保守性評価枠組み

A Framework For Evaluate Software Maintenability In Consideration Of The Abstraction Level In Design Phase

志水 理哉*　鷲崎 弘宜†　深澤 良彰‡　伊藤 弘毅§　田邉 浩之¶　波木 理恵子‖

あらまし　近年のソフトウェアは要求を満たすだけでなく保守がしやすいことが求められる. ソフトウェアの保守性は主に設計段階で作りこまれる. そのため, 早い段階からの品質管理は重要であるが早い段階の成果物は企業や現場によって異なる抽象度で生成されるため, 統一の評価基準で品質管理を適切に行うことは困難である. そこで我々はソフトウェア設計クラス図を対象に, 抽象度を考慮した品質管理手法を提案する. 本論文では ET ソフトウェアデザインロボットコンテストで提出された設計クラス図を対象に評価実験を行った. 評価実験では本手法の抽象度による層別の有用性を示し,2 つのモデルに対しケーススタディを行った.

1　はじめに

近年のソフトウェアは大規模かつ複雑化していることに加え短納期が求められる. 短納期を実現するために多くの現場では全ての機能を新規に開発するのではなく既存のソフトウェアの改良, 派生によって開発している. もし, 既存のソフトウェアが保守しにくい物であれば短納期を実現することが難しくなる. そのため近年のソフトウェアには保守性が高いことが望まれる. 実装段階で保守性を向上させることは難しく, 高い保守性を維持するためには保守性の品質低下要因が作りこまれる設計段階から品質管理を行っていくことが必要である. 実装段階の品質管理に関しては様々な自動化手法 [1] [2] が研究されているが，設計段階に関しては，多くの現場において人手のレビューによる品質管理が行われている. 人手による品質管理では抜けや漏れが発生するリスクに加え，実施者ごとに評価のずれが生じる恐れがある. 設計段階の自動評価が普及していないのはソフトウェアモデルの評価枠組みが確立されていないことと厳密な抽象度に関する決まりがないためである. ソフトウェアモデルは利用目的ごとに様々な抽象度で表現されている. また利用目的は同じであっても開発の環境によって生成されるソフトウェアモデルの抽象度は統一されていない.

我々は設計段階における抽象度の違いを考慮するため 3 段階に層別することを提案し，そのためのチェックリストを考案する. そして，ISO9126 [3] を基にした様々な側面の品質特性を設計段階で評価するため，設計モデルを対象とした品質メトリクススイートの考案を行う. 本論文ではソフトウェア設計クラス図を対象とした保守性の評価について述べる. 設計モデルの抽象度を考慮して評価することで精度の高い品質管理が可能となる.

*Masaya Shimizu, 早稲田大学
†Hironori Washizaki, 早稲田大学
‡Yoshiaki Fukazawa, 早稲田大学
§Hiroki Itoh, 早稲田大学
¶Hiroyuki Tanabe, 株式会社オージス総研
‖Rieko Namiki, 株式会社オージス総研

2　関連研究

ISO9126 ではソフトウェアの品質特性を 6 つ定義し，それぞれの品質特性の配下に複数の品質副特性を定義している．組み込みソフトウェアの品質を定量的に測定する Adqua [12] ではソフトウェアの最終成果物であるソースコードを対象にそれらの品質を評価する．しかし，品質低下の原因のいくつかは設計段階で作りこまれる．Genero ら [9] や佐藤ら [6] は設計段階の成果物である設計モデルを対象にメトリクスを測定し品質評価を行った．Cruz [7] はソースコードメトリクスを用いた評価方法を設計モデルから測定可能なメトリクスにあてはめることを試みた．その際にソースコードと設計モデルでは同じメトリクスでも得られる値の尺度が異なることを指摘し, 尺度をそろえるため正規化の必要性を述べている．

設計モデルは主に UML [11] で記述される.UML はソフトウェアモデルの統一された記述方法を定義しているが, Lange ら [5] の研究ではソフトウェアモデルの利用目的がソフトウェアの開発フェーズごとに異なることため, ソフトウェアモデルには同じダイアグラムでも利用目的にあわせた異なる抽象度の成果物が存在することを述べ, それぞれの抽象度に合わせた品質管理の手法を提案している．しかし実際は開発フェーズ, 利用目的が同じでも抽象度が異なるソフトウェアモデルが存在する．

設計段階の成果物からメトリクスを測定し, いくつかのメトリクスはソースコードメトリクスと同様にソフトウェアの品質評価に用いることは可能である．しかし我々は設計段階では異なる抽象度で記述された成果物の存在を考慮し, 全ての設計モデルを統一の方法で高精度に評価することは困難であると考える．そこで我々は設計段階のモデルを対象とし，抽象度を層別し，各抽象度の特性を考慮した評価をすることを検討する．

3　ソフトウェア設計クラス図を対象とした保守性評価枠組み

我々は設計クラス図を対象に保守性の評価を行うために, 設計クラス図から測定可能なメトリクスを挙げそれらを用いる．設計クラス図からメトリクスを測定し, それぞれのメトリクスの metrics 値が保守性にどのように影響しているのかを見たいときに, それぞれの metrics 値の尺度が異なると複数のメトリクスを用いて評価することが難しい. そこでそれぞれの metrics 値を 0 から 100 の得点として正規化するために正規化式を求める. しかし, 設計クラス図の抽象度が異なれば,metrics 値の傾向もことなるため異なる抽象度で統一の正規化式を用いることは望ましくないと考える．そこでまず, 複数のソフトウェア設計クラス図を抽象度で層別し, それぞれの抽象度に応じた評価を行うための正規化式をもとめる．そして設計クラス図を対象とした保守性の評価では評価対象の設計クラス図の抽象度に対応した正規化式を用いて評価を行う. それぞれの流れを図 1,2 に示す.

3.1　抽象度の層別

設計段階の設計クラス図は主に実装を行うために生成されるが, そこに含まれる情報量は開発の現場によって様々である．例えば図 3,4 は同じ組み込みロボットが提供する API について書かれているが図 4 に対し図 3 の方が情報量は少なく抽象度は高く記述されている．このソフトウェアモデルの抽象度について一般的な取り決めはなく，そのため開発の現場ごとに様々な抽象度の設計モデルが利用されている．抽象度を考慮せず, 全ての設計モデルに対し同じ評価基準で評価を行ってしまうと, 評価基準を満たさなかったときに, それは開発者の設計が悪いのか抽象度の問題なのかがわからない．そこで我々はまず抽象度を考慮できる仕組みを提案する．

我々は設計クラス図の抽象度を定量的に層別するため, クラス図を構成する各要素に着目した. 例えば,「関連」に着目すると「有向関連」になっているものの方が, 向きが表記されていないものと比較し情報量は多く, 抽象度は低いと考えられる. そこで「すべての関連が有向関連になっている」という条件を満たしている設計クラ

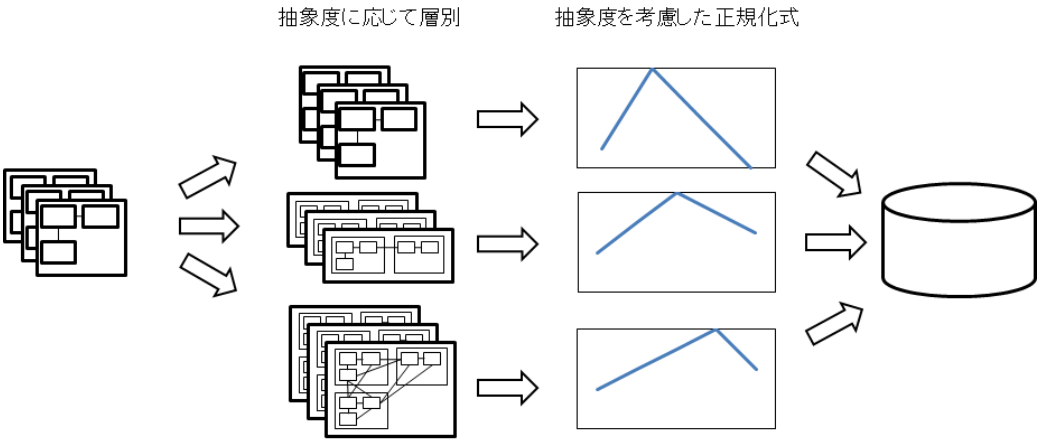

図 1　Metrics 値の正規化式を求める

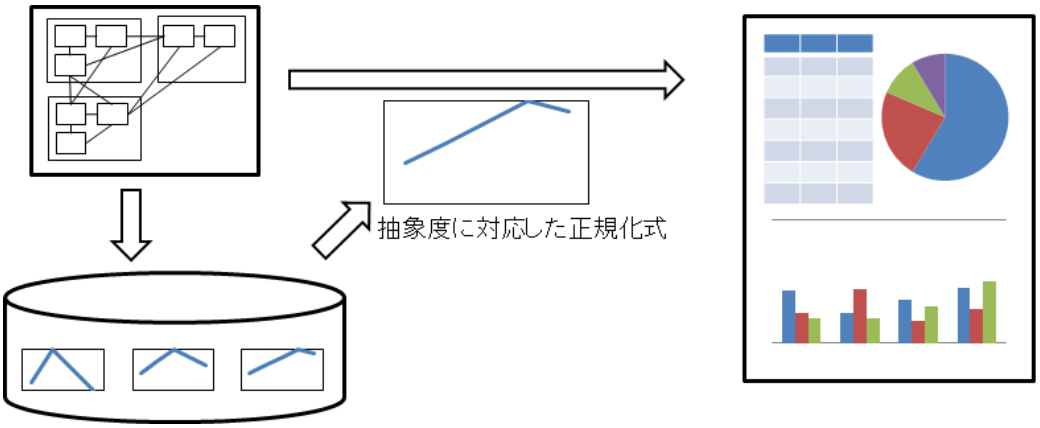

図 2　ソフトウェア設計クラス図を評価する

ス図は満たしていない設計クラス図よりも抽象度は低いとした. その他の要素についても同様に検討し我々は表 1 に示す抽象度を層別するためのチェックリストを考案した.

チェックリストを満たす数が 0 個から 3 個を抽象度低，4 個から 6 個を抽象度中，7 個から 10 個を抽象度低と層別する．また，抽象度高と層別された設計モデルは品質を評価するのに十分な情報が得られない場合があるため本研究では評価対象外とする.

3.2　保守性評価メトリクススイート

我々は ISO9126 の品質特性の中で設計クラス図と関係が深い保守性に焦点を当てる. 本研究では保守性の品質副特性である解析性, 変更性を設計クラス図から評価する手法を提案する．我々は解析性・変更性を様々な側面から評価するため GQM 法 [4] を用いて品質メトリクススイートを考案した. Goal 層を解析・変更性，Question 層を着目すべき側面,Sub Question 層を着目すべき要素,Metrics 層を設計クラス図から抽出可能なメトリクスとした. GQM 法を用いることで,Goal 層が満たされなかった場合 Question 層,Metrics 層と任意の粒度でさまざまな側面から問題点を検討することが可能である．表 2 に解析・変更性の Goal - Question - Sub Question を示す.

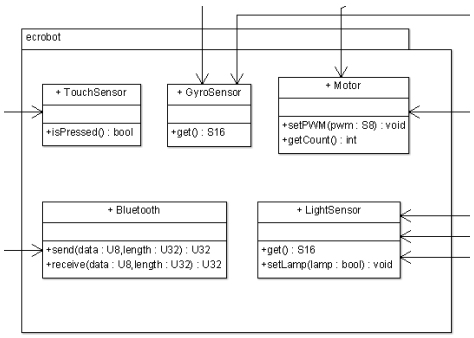

図 3　抽象度の高いクラス図

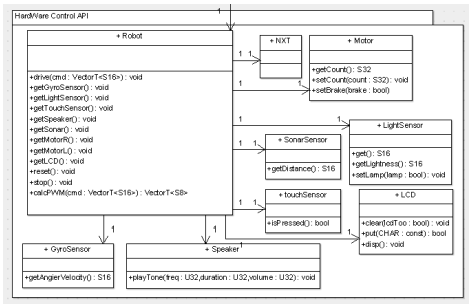

図 4　抽象度の低いクラス図

表 1　抽象度層別のためのチェックリスト

構成要素	チェック項目	備考
パッケージ	パッケージを考慮しているか	設計モデル上でパッケージが一つ以上あるか.
クラス	全てのクラスにアクセス可能なクラスがあるか	全てのクラスに可視性が public な属性か操作が一つ以上あるか.
属性	属性の型が検討されているか	属性の型に環境依存，システム依存な内容が現れるか.
属性	属性の可視性が検討されているか	可視性が public な属性，private な属性がそれぞれ 1 つ以上存在しているか.
操作	操作の引数が検討されているか	操作の引数が検討されている要素があるか.
操作	操作の戻り値の型が検討されているか	操作の戻り値の型に環境依存，システム依存な内容が現れるか.
操作	操作の可視性が検討されているか	可視性が public な操作，private な操作がそれぞれ 1 つ以上存在しているか.
関連	全ての関連が有向関連になっているか	全ての関連が有向関連になっているか.
関連	コンポジット関連，集約関連があるか	コンポジット関連や集約関連が現れるか.
汎化・実現	階層化が検討されているか	汎化，実現の関係が現れるか.

解析・変更性の評価は全体の可読性，構造，階層化の三つの側面を見る. その中で例えば全体の可読性について評価を行うには Sub Question の「全体のサイズは適切か」と「読みにくい表記をしていないか」を見る. Sub Question を評価するための Metrics のうち解析・変更性の評価に利用するものを表 3 に示し，Sub Question と Metrics の対応を表 4 に示す. Metrics はシステム単位で測るもの (S), パッケージ単位で測るもの (P), クラス単位で測るもの (C) がある. 例えば，「全体のサイズは適切か」を評価するにはシステム単位で測る「S001:パッケージ数」とパッケージ単位で測る「P001：クラス，インタフェース数」のメトリクスを用いて行う. これらのメトリクススイートは複数の実務者，専門家へのインタビューの繰り返しを通じ考案したものである.

3.3　メトリクス値の正規化

提案手法において測定された metrics 値を 0 から 100 の得点として正規化を行う. 複数のサンプルモデルを対象に metrics 値を測定，分析しヒストグラムの形を参考に最頻出値を 100 とする正規化式を各メトリクスに用意する. 例えばあるメトリク

表 2　解析・変更性の Goal - Question - Sub Question

Goal	Question	Sub Question
解析・変更性	全体の可読性は良いか	全体のサイズは適切か
		読みにくい表記をしていないか
	構造は良いか	クラス間の結合度は低くなっているか
		パッケージ間の結合度は低くなっているか
		カプセル化を考慮しているか
		責務の量は適切か
		要素の凝集度は高くなっているか
	階層化は良いか	共通部分と可変部分を分離しているか
		階層化の形状は適切か

表 3　Metrics とその概要

Metrics	略称	詳細
要素名の単語数	C001	クラス名，インタフェース名の単語数
結合しているクラス数	C002	結合しているクラス数
結合されているクラス数	C003	結合されているクラス数
属性数	C004	クラスが持つ属性（継承した属性も含める）の数
操作数	C005	クラスが持つ操作（継承した操作も含める）の数
継承した属性の割合	C006	継承した属性の割合（継承した属性数/全属性数）
継承した操作の割合	C007	継承した操作の割合（継承した操作数/全操作数）
子クラスの数	C008	対象クラスを拡張・継承するクラスの数
親クラスの数	C009	対象クラスが拡張・継承しているクラスの数
要素数	P001	クラス，インタフェースの数
パッケージ名の単語数	P002	パッケージ名の単語数
結合しているパッケージ数	P003	結合しているパッケージ数
結合されているパッケージ数	P004	結合されているパッケージ数
パッケージ外から結合されている要素の数	P005	パッケージ外のクラスから結合されている要素の数
凝集度の欠如	P006	パッケージ内の (存在していない関連の数)-(存在している関連の数)
親クラスがないクラスの数	P007	パッケージ内の親クラスを持たない基底クラスの数
子クラスが他から直接関連されている数	P008	パッケージ内のクラスで親クラスを持つものに結合しているクラスの数
多重継承している直下のクラスの割合	P009	パッケージ内に含まれるクラスで多重継承しているクラスの割合
継承木の深さ	P010	パッケージ内で最大の継承木の深さ
パッケージ数	S001	パッケージ数
全クラス間の Coupling Factor	S002	全クラス間の結合度（実在する関連数/想定される最大の関連数）
全パッケージ間の Coupling Factor	S003	全パッケージ間の結合度（実在する関連数/想定される最大の関連数）
多重継承しているクラスの割合	S004	全クラスで多重継承しているクラスの割合

スにおいて図 5 のようなヒストグラムが得られたとする. メトリクスによっては図のように抽象度によって Metrics 値の頻度の傾向が異なる．そこで我々は正規化式を層別した抽象度ごとに分けて用意することにした．図 5 のヒストグラムによってえられる正規化式が図 5 である．ヒストグラムで最頻値の値を 100 とし, ヒストグラムの形を考慮して正規化式の傾きを決定する. 以上のように設計クラス図を抽象度で層別し，その抽象度に特化した正規化式を用意することで品質評価結果もそれぞれの抽象度に特化したものになり，本手法の精度が向上することが期待できる.

表 4　解析変更性の Sub Question - Metrics

Sub Question	Metrics
全体のサイズは適切か	S001,P001
読みにくい表記をしていないか	P002,C001
クラス間の結合度は低くなっているか	S002,C002,C003
パッケージ間の結合度は低くなっているか	P003,P004,S003
カプセル化を考慮しているか	P005
責務の量は適切か	C004,C005
要素の凝集度は高くなっているか	P006
共通部分と可変部分を分離しているか	C006,C007,P007
階層化の形状は適切か	C008,C009,P008,S004,P009,P010

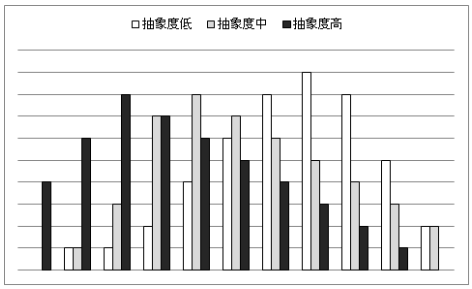

図 5　metrics 値のヒストグラム

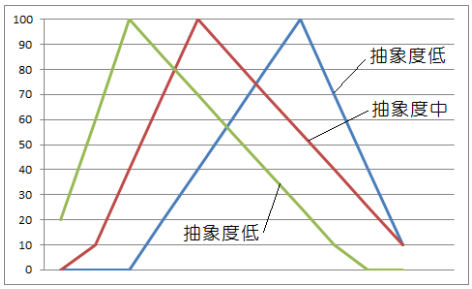

図 6　ヒストグラムから得られる正規化式正規化式

4　評価実験

我々は ET ソフトウェアデザインロボットコンテストで提出された設計モデルを対象に，評価実験を行った．実験は 50 の設計モデルを対象に抽象度グループを判別し，抽象度低，抽象度中のグループの正規化式を決定した．抽象度高を対象から外したのは評価するのに必要な要素が十分でなかったためである．今回用いた 50 の設計モデルのうち，9 が抽象度低，10 が抽象度中，31 が抽象度高であった．

4.1　正規化式

表 5, 図 7,8 は「Question：構造は良いか」に対応するクラス単位のメトリクスについて 50 の設計クラス図を対象に分析したものである. 平均値を見ると抽象度が高くなるにつれ値も大きくなっている. これは抽象度が低いとより詳細に記述されているため Metrics 値が大きくなったと考えられる. 標準偏差を見ると抽象度中, 抽象度低の値は全体を対象とした時と比較して値が小さい. このことから我々の抽象度の層別は抽象度中, 抽象度高の設計クラス図において抽象度による Metrics 値のばらつきを抑えることに有用である. 抽象度低の設計クラス図では標準偏差は逆に大きくなっている. これは詳細に記述されたことにより各設計クラス図の製作者の特色が強く出たためと考える.

以上のことを踏まえると,Metrics 値は抽象度によって傾向が異なるので,Metrics 値の正規化式もそれぞれの抽象度で異なるべきである．今回我々は抽象度低, 抽象度中を対象に正規化式を求めた. 図 9,10 は属性数 (C004)，操作数 (C005) について Metrics 値とその値の頻度が全体に占める割合を抽象度で比較したものである．図 11,12 に属性数 (C004)，操作数 (C005) の正規化式を示す.

表 5　抽象度による Metrics 値の平均と標準偏差

	平均値				標準偏差			
	全体	抽象度低	抽象度中	抽象度高	全体	抽象度低	抽象度中	抽象度高
C002	0.86766	1.35433	0.99115	0.57905	0.09881	0.16600	0.06593	0.02527
C003	1.47264	1.70472	1.35841	1.40952	0.04818	0.08127	0.02384	0.02577
C004	0.81891	1.11024	0.79204	0.68952	0.03726	0.05583	0.05269	0.01355
C005	1.52736	2.64961	1.57080	0.96571	0.07800	0.08473	0.03797	0.00150

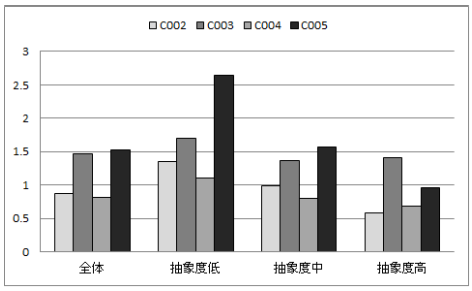

図 7　抽象度による Metrics 値の平均値比較

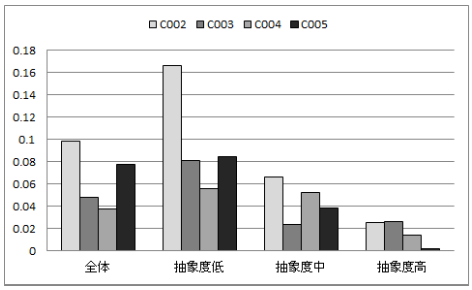

図 8　抽象度による Metrics 値の標準偏差比較

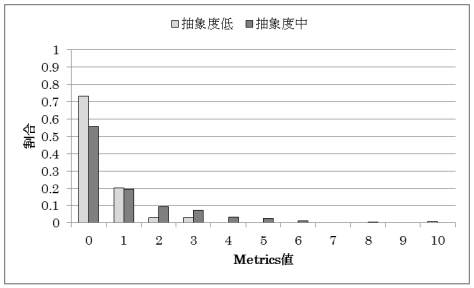

図 9　属性数の傾向の比較

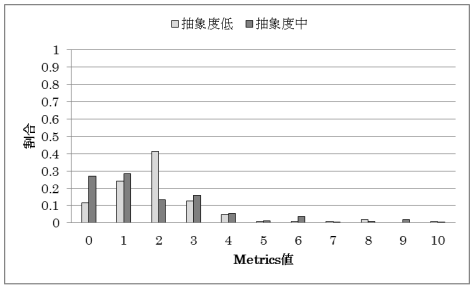

図 10　操作数の傾向の比較

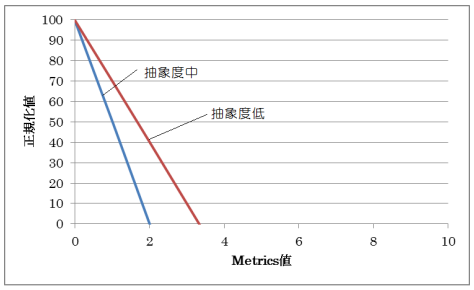

図 11　属性数の正規化式

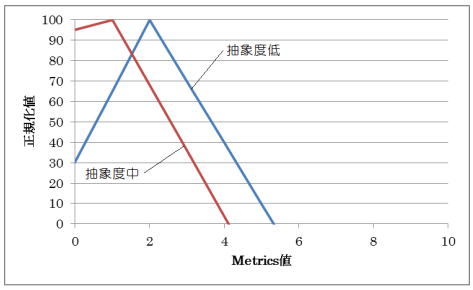

図 12　操作数の正規化式

4.2　品質評価

我々は考案したメトリクススイートと正規化式を用いて実際に ET ロボットコンテスト 2010 で提出されたモデルの評価を行った. 評価対象とした設計クラス図を図 13,14 に示す. このモデル A,B の審査員によるモデル評価では構造に関する項目でそれぞれ A 評価,C 評価を受けている.

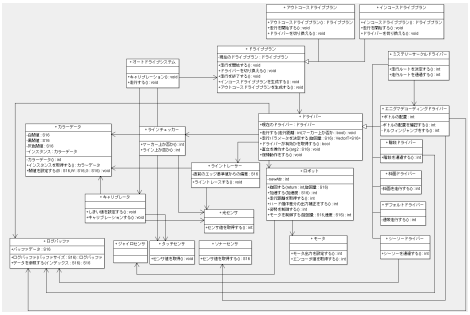

図 13　評価対処モデル A

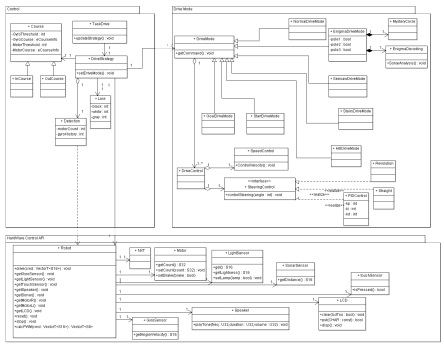

図 14　評価対象モデル B

表 6 にモデル A の Question 層,Sub Question 層の評価結果を示す. まず, 対象の設計クラス図ではパッケージが考慮されていなかった. そのため Sub Question 層,Metrics 層のいくつかは測定対象外となっている. 「Sub Question：全体のサイズは適切か」の得点が極めて小さいのはパッケージが考慮されていないことが大きな要因である. 全ての要素が 1 つのパッケージ内にあると見ることができるが, そのパッケージのサイズはきわめて大きなものとなる. その結果として得点はきわめて低いものとなった. 同様に「SubQuestion:要素の凝集度が高くなっているか」もパッケージが考慮されていないことで関係の薄い要素でも同一パッケージ内に存在していると見ることができる. よって点数はきわめて低いものとなっている.

表 7 はモデル B の評価結果である. モデル B ではパッケージが考慮されておりモデル A よりも可読性は良くなっていると考えられる. しかし,1 パッケージ内に含まれる要素の数が多いため高い点数にはなっていない。さらに可読性を向上させるためには各要素の役割を整理し統合できる要素は統合し要素数を減らすか, パッケージを増やしてより細かく要素を分類する改善策が挙げられる.「SubQuestion:要素の凝集度が高くなっているか」の得点が低くなっていることからもこのモデルにおいてパッケージ分割の再考の必要性があることが予想される. 一方, このモデルでは Robot クラス,DriveStrategy クラス,DriveMode クラスを中心として各要素が関連しているため, 各要素の結合関係は整理されている. そのため, 結合度に関する項目において高い得点となっている.

表 6　Question 層-Sub Question 層の評価結果 モデル A

Question	得点	Sub Question	得点
全体の可読性は良いか	50.00	全体のサイズは適切か	0.00
		読みにくい表記をしていないか	100.00
構造は良いか	59.79	クラス間の結合度は低くなっているか	67.24
		パッケージ間の結合度は低くなっているか	undef
		カプセル化を考慮しているか	undef
		責務の量は適切か	71.92
		要素の凝集度は高くなっているか	0.00
階層化は良いか	52.03	共通部分と可変部分を分離しているか	20.73
		階層化の形状は適切か	83.33

今回の本手法による評価ではモデル B の方がモデル A よりも高い値を示している. このことは審査員の評価結果と矛盾している. このような結果となった一番の理由はモデル A にパッケージの考慮が欠けていたことと考える. モデル B では要素を

表 7　Question 層-Sub Question 層の評価結果 モデル B

Question	得点	Sub Question	得点
全体の可読性は良いか	60.89	全体のサイズは適切か	21.79
		読みにくい表記をしていないか	100.00
構造は良いか	75.74	クラス間の結合度は低くなっているか	88.98
		パッケージ間の結合度は低くなっているか	74.36
		カプセル化を考慮しているか	100.00
		責務の量は適切か	80.36
		要素の凝集度は高くなっているか	35.00
階層化は良いか	54.17	共通部分と可変部分を分離しているか	18.75
		階層化の形状は適切か	89.58

パッケージで 3 層に分割し, それぞれのパッケージ間の関連を限定しているため, モデル A よりも構造面で優れていると我々は考察する. 審査員評価でモデル B の評価が悪くなった原因として審査員よりクラスの抽出方法が指摘されていた. このことはクラス名を見ることで判断できるが, 我々の手法ではクラス名に関して単語数を測定しているだけであるため現状では判断できない. すなわち自然言語を対象とした部分の評価に関しては本研究では扱えておらず, 人手による評価が必要である. しかし要素間の関連など設計クラス図の構造に関する部分は定量的に抜けや漏れなく属人性を排して評価が可能である. よって現状では我々の手法と人手による従来通りのレビューによる評価を併用するのが望ましいと考える.

5　おわりに

本研究では設計段階を対象とした品質評価のためにメトリクススイートを考案した. また, それを利用した評価を抽象度に応じて評価ができるよう抽象度の層別とその方法を提案し, Metrics 値の正規化式を抽象度ごとに設定することを述べた. これによりそれぞれの抽象度に特化した評価が可能になる. 4 章では ET ソフトウェアロボットコンテストで提出された設計モデルを対象に本研究で考案した抽象度の層別の有用性を示し, また考案したメトリクススイートと Metrics 値の正規化式を用いてロボットコンテストで提出された 2 つの設計クラス図に対し評価を行った. その結果それぞれのモデルで再検討すべき項目を挙げることができた.

今回評価実験で用いた 2 つの設計クラス図の評価結果はでは ET ソフトウェアロボットコンテストの審査員評価結果と異なっていた. 我々の手法では属性の数や関連の数など定量的なメトリクスに着目し測定を行っているのに対し, 審査員の評価ではクラス名や属性名などの意味や表現に着目している傾向があると考えている. このように着目している箇所が異なっていたことが評価結果の乖離につながったと考えている. 現在の我々の手法では言葉の意味や表現を評価対象にできてはいない. しかし要素間の関連など設計クラス図の構造に関する部分は定量的に抜けや漏れなく属人性を排して評価が可能である. 現状では本手法と人手によるレビューを併用されることが望ましいが, 将来的には言葉の意味や表現を評価対象とし定量的に測定することを視野に入れている.

また我々は本手法でより高精度に評価をするため Metrics 層,Sub Question 層に重み付けを行うこと, 正規化式の求め方を各メトリクスごとに再検討することを考えている. 現在,Metrics 層,Sub Question 層で重み付けを行っていないため, それぞれの項目が評価結果にもたらす影響は等しくなっているが, 実際は項目ごとに品質に与える影響は異なるはずである. 品質に大きく影響のあるものの重みは大きく, 影響の小さなものの重みは小さく設定し評価ができるようになることで本手法はより受け入れられやすくなると考えている. また, 正規化式は, 全て最頻値を 100 とする一次関数とした. より品質評価の精度を上げるためには例えば一部の正規化式は指数

関数を用いるなど, メトリクスごとにそのメトリクスに特化した正規化式の求め方が必要である．現在の我々の手法では扱えていない自然言語の部分の評価,Metrics値が品質特性に与える影響をより高精度に捉えることができるようになることで本手法全体の精度が向上し，より有用なものにできると考えている．

謝辞 本研究に際して,ET ロボットソフトウェアコンテストで得られた資産を活用させていただきました. コンテスト関係者の皆様に感謝いたします.

参考文献

[1] Hironori Washizaki, Hirokazu Yamamoto, Yoshiaki Fukazawa:A Metrics Suite for Measuring Reusability of Software Components, In Proceedings of IEEE METRICS'03, pp.211-211, 2003.
[2] 森 俊樹, 櫻庭 紀子, 中野 隆司:ソフトウェア品質技術の開発と適用, 東芝レビュー, Vol.61, No.1, pp.26-31, 2006.
[3] 東 基衛 編:ソフトウェア品質評価ガイドブック, 1994.
[4] Victor R.Basili, David M.Weiss:A Methodology for Collecting Valid Software Engineering Data, EngineeringData, IEEE Trans. on SoftwEng., Vol.SE-10, No.6, pp.728-838, 1984.
[5] Chiristian F. J. Lange, Michel R:Managing Model Quality in UML-based Software Development, Proc. Of the 13th Int. Workshop on Software Technology and Engineering Practice, 2005.
[6] 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, 情報処理学会論文誌, vol.49, N0.7, pp.2319-2327, 2008.
[7] Ana Erika Camargo Cruz:Exploratory Study of a UML Metrics for Dault Prediction, In Proceedings of International Conference on Software Engineering*ICSE'10*, Vol2,pp.361-364,2010.
[8] Chen, Yue and Boehm, Barry W. and Madachy, Ray and Valerdi, Ricardo:An Empirical Study of eServices Product UML Sizing Metrics, Proceedings of the 2004 International Symposium on Empirical Software Engineering*ISESE'04*, pp.199-206, 2004
[9] Marcela Genero, Mario Piatini:Finding "early" indicators of UML class diagrams understandability and modifiability, In Proceedings of International Symposium on Empirical Software Engineering*ISESE'04*, pp.207-216, 2004.
[10] Ma Esperanza Manso, Marcela Genero, Mario Piattini:No-redundant metrics for UML class diagram structural complexity, In Proceedings of international conference on Advanced information systems engineering*CAiSE'03*, pp.127-142, 2003.
[11] Object Management Group. Unified Modeling Language, *http://www.uml.org/*.
[12] 株式会社オージス総研. Adqua, *http://www.ogis-ri.co.jp/product/b-08-000001A6.html/*
[13] J.Wust. The software design metrics tool for the UML, *http://www.sdmetrics.com/*.