# 利用関係の一致に基づくソフトウェア部品のクラスタリング手法
## ライブラリ部品の利用も考慮した手法の実現

2018SE021 石井敦士　2018SE025 神田晃希　2018SE050 村山健人
指導教員：横森励士

## 1　はじめに

ソフトウェアを構成する部品の数は増大しており，ソフトウェアを把握するために，ソフトウェア部品の類似性を用いて，部品群として提示する手法が有効であると考えられる．[2] では，ソフトウェア部品がどの部品を利用しているかに基づいて，部品対ごとに類似性を計算し，階層的クラスタリングを行ってソフトウェアを分類する手法が提案されている．さらに，ソフトウェア内で定義された部品の利用状況の一致度，ソフトウェア外で定義された部品の利用状況の一致度から類似度を求め，それらを組み合わせ，階層的クラスター分析を行う手法 [4] が提案されている．[3] では精度向上を目的として，共通利用部品数に基づいて分類を行う手法が提案されているが [4] ではまだ試すことができていない．本研究では，[3][4] の手法を組み合わせることで考えられる部品の分類手法について，得られる部品群がどのように変化するかを調査する．ソフトウェア内外で定義された部品の利用状況を考慮して分類を行う手法について，一致度，一致部品数による類似度計算手法による部品分類手法の組み合わせのバリエーションを提案する．それぞれの部品分類手法の結果がどのように変化するかを調査し，それらの違いが精度の向上に繋がるかを調査する．提案手法をソフトウェア理解を目的とした手法で用いる際にどのように活用できるかを考察する．

## 2　背景技術

### 2.1　ソフトウェア部品と部品グラフ

ソフトウェア部品とは，その内容をカプセル化したうえで，ソフトウェアを実現する環境において交換可能な形で配置できるようにしたシステムのモジュールの一部を指す．本研究では，各クラスのソースコードを記述しているファイルを部品とみなし，各部品を構成要素とする部品グラフを構築する．部品グラフ上の頂点は各部品を表し，辺は部品間の関係として利用関係を表現する．ある部品 A が他の部品 B を利用している場合，A から B への利用関係が存在するとみなし，A から B への有向辺で表現する．

### 2.2　関連研究

ソースコードからパターンを抽出し，ソフトウェア理解支援に活用する研究が行われている．Zhong らは [1] で，ソースコードから API の利用順を抽出し，API の利用方法の学習に活用するシステムを提案した．横森らは，ソフトウェア部品間の利用関係の一致度から各部品間の類似度を求め，距離行列を計算し，階層的クラスタリングによって樹形図を作成し，樹形図から類似部品群を得るというソフトウェア部品を分類する手法を提案した [2]．[2] での評価実験では，得られた部品群内の部品の多くが類似度を持つことが確認されており，ソフトウェアの部品の利用関係の観点におけるある一つの切り口から類似度を抽出した手法になっているため類似性の判断には不十分であると考えられる．藤田らは，利用部品の一致度ではなく，共通利用部品数に基づいて類似度を計算する手法を精度向上の手法として提案し [3]，一部の部品の結合の仕方が変わり，類似性を持つ部品が集約されやすくなったことを示した．青山らは，ソフトウェア内で定義された部品の利用状況の一致度，ソフトウェア外で定義された部品の利用状況の一致度から類似度を求め，それらを組み合わせて階層的クラスター分析を行う方法を提案し，それぞれ単独で分類した結果と比べて，分類対象となる部品数が増えたことを示した [4]．分類結果を精査したところ，ソフトウェア内で定義された部品の利用状況の一致度についての分類では，ソフトウェア内の観点から分類されることが多く，一方で，ソフトウェア外で定義された部品の利用状況についての分類では，ソフトウェアの部品が実際に行っている機能の観点から分類されることが多く，手法によって分類結果の意味が異なることがわかった．[2] の手法を活用する方法として，コードブラウザなどで部品を把握する際に，利用関係などの情報とともに，類似した部品の情報もあわせて提示することで，ひとまとめにして理解できるような部品の情報を提示でき，理解の効率を高めることができると我々は考えている．その際に，[3][4] などの精度向上の手法や，いろいろな観点からの分類が利用できると考えている．

## 3　ソフトウェア内の部品の利用関係に基づく分類手法における類似度計算手法

### 3.1　研究の動機

[3][4] などの精度向上を目的とした手法は個々に考えられているが，それらを組み合わせたときに理解の効率の向上に役立つかどうかはまだ検証されていない．実際のプロジェクトに対して組み合わせた手法をそれぞれ適用することで，どのように得られる部品群の違いがでるか，その違いが何を表すかについて確認する必要がある．

### 3.2　研究の内容

本研究では，ソフトウェア内外の部品の利用状況を考慮して分類を行う手法を実現する際に，どのようなアプローチが適切であるかを考察するために，[3][4] の手法の組み合わせた部品分類手法のバリエーションを提案し，それぞ

れの手法によって導出された部品群にどのような違いがでるかを調査する．さらに，それらの違いが精度の向上を意味するのか，異なる観点から類似していることを示しているのかを調査し，どのようなアプローチが適切なのかを調査する．これらによって，[3][4] などの手法を理解できる環境でどのように活用すべきかを考察する．

### 3.3 ソフトウェア部品の利用状況に基づく分類手法のバリエーションについて

本研究では，ソフトウェア部品の利用状況に基づいてクラスタリングを行う手法について，類似度を計算する手法を複数用いるので，それらの違いを説明する．

一つ目の観点は，利用部品の集合の定義である．

- ソフトウェア内で定義された部品の利用を用いる．
- ソフトウェア外で定義された部品の利用を用いる．

の違いで，それぞれ（内…）（外…）で表記する．

二つ目の観点は，類似度の計算方法である．

- 集合の類似度である Jacard 係数で類似度を表す．
- 利用部品の一致数によって，類似度を表す．

の違いで，それぞれ（…一致度）（…一致数）で表す．

[3] は（内一致数）に基づいた分類手法で，[4] は（内一致度）と（外一致度）の平均で類似度を求めている．要旨では [3][4] の手法の組み合わせとして，

- （内一致数）と（外一致数）の類似度の平均
- （内一致数）と（外一致度）の類似度の平均
- （内一致度）と（外一致数）の類似度の平均
- （内一致度）と（外一致度）の類似度の平均

の四つの組み合わせでそれぞれ距離行列を作成し，得られた部品群の違いを評価する．具体的な計算方法を示す．

1. 共通利用部品数の一致度に基づいた分類

   類似度 sim(A,B) と距離 dist(A, B) を以下のように定義する．分析対象のソフトウェア部品の集合を C とする．ある部品 A の利用先部品の集合を $O_A$，ある部品 B の利用先部品の集合を $O_B$ とする．

$$sim(A,\ B) = \frac{|O_A \cap O_B|}{|O_A \cup O_B|}$$

$$dist(A,\ B) = 1 - sim(A, B)$$

2. 共通利用部品数の一致数に基づいた分類

   一致利用部品数 (com) と部品間の相違度 (diff) を以下のように定義する．$C$ の部品の全部の組み合わせにおける $com(X \in C,\ Y \in C)$ の最大値を $MAX(C)$ とする．

$$com(A,\ B) = |O_A \cap O_B|$$

$$diff(A,\ B) = 1 - \frac{|sim(A,\ B)|}{MAX(C) + 1}$$

### 3.4 類似性の基準について

得られた樹形図それぞれにおいて，類似した部品のまとまりを調査し，類似部品群を求める．部品間の関連性を確認する際には，以下の 4 つの基準を判定の基準とする．

1. 部品の役割が同じである．ある 1 つの機能を実現する部品でまとまり，同じ目的の処理が行われている部品として関連していると判断する．
2. 部品の扱う対象が同じである．必要な前処理や後処理が一致しているなど同種の処理が行われている部品として関連していると判断する．
3. 同じパッケージに所属している．開発者が同種の目的と判断し，パッケージ構造の中で部品の整理を行った結果と一致していると考える．
4. ファイル名に関連がある．ファイル名には，そのファイルの機能を明示するという慣習があるので，ファイル名の一部が一致していることで類似した役割を持つと考えられる．

## 4 評価実験

### 4.1 評価実験の概要

実際のオープンソースソフトウェアを対象とし，3.3 で示す 4 種類の類似度計算手法を用いてそれぞれ樹形図を作成し，ソフトウェア部品を分類する．得られた 4 通りの部品群に対して，以下の違いを調査する．

- 観点 1　得られた類似部品群に存在する部品数がどう変化したか．分類結果として得られた部品群内の部品数を比較し，類似性を持つと判定できる部品の数についての傾向を求める．
- 観点 2　得られた類似部品群一つあたりの部品数がどのように変化し，遷移したか．部品群 1 つあたりの部品数を比較し，大きな役割でまとまるのか，細かくまとまる傾向があるのかを調査する．
- 観点 3　得られた類似部品群の中身がどのように変化したか．手法の違いにより，部品の所属する部品群がどう変化したかを，表を作成して，比較する．

### 4.2 クラスター抽出の手順

1. 分析対象のソフトウェアを Classycle と CCfinder で分析して，各部品の情報や利用関係を入手する．
2. 各部品の利用部品（内，外）をまとめる．
3. 3.3 の類似度計算手法で，類似度と距離を計算する．
4. 各部品間の距離を示す距離行列を作成する．
5. 階層的クラスター分析を行い樹形図を得る．
6. 得られた樹形図において，葉の部分からまとまりになっている部分を抽出し類似部品群を得る．

### 4.3 実験結果

jlgui と barbecue の二つの java アプリケーションを対象に実験を行った．jlgui はイコライザ機能を搭載した MP3 プレーヤーで，ver3.0 は 70 のソースファイル（部品）で構成されている．barbecue は国際規格に適合したバーコードを作成するソフトウェアで，ver1.5 は 59 のソースファイル（部品）で構成されている．最初に jlgui を対象とし，3.3 で得た 4 つの類似度から樹形図を求め，比較する．（内一致度）と（外一致度）の類似度の平均で分類したときの樹形図を図 1 に示す．結果として 17 個の部品群が形成され，総部品数 70 個のうち 58 個が部品群に分類され，残り 12 個が分析対象外であった．

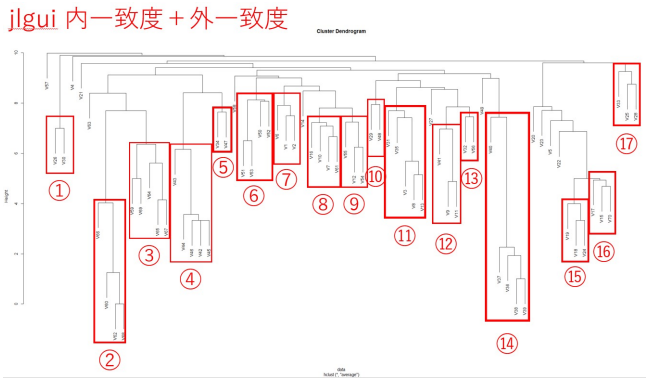


図 1 （内一致度）と（外一致度）の類似度の平均で分類したときの樹形図

それぞれの手法で樹形図を求め，部品群を抽出した結果を表 1 に示す．4 つの手法で，類似度の計算方法が変わっても所属部品群が変わらない部品は 45 個で，いずれの手法でも分析対象外だったのは 4 個あった．残りの 21 個は計算方法によって結果が変わった．8 個はいずれの場合も類似性を確認できず，13 個はいずれかの結果で類似性を確認できた．表 3 のように分類先がどのように変化したかを調査したところ，7 個は 1 つの部品群で類似性を確認でき，6 個は複数の部品群で類似性を確認できた．例えば，部品 ID20 の ActiveJlabel という部品では (内一致数＋外一致度) の時，ActiveJIcon,ActiveJNumberLabel と同じ部品群に所属しており，類似性の基準 1,2,3,4 を満たしていた．また，（内一致度＋外一致度），（内一致度＋外一致数）の時，ActiveJButton,ActiveJToggleButton となり，部品群が変わったが，基準 1,2,3,4 を満たしている．

次に barbcue の ver1.5 を対象として，同様に 4 つの類似度から樹形図を求め，比較した．部品群を抽出した結果を表 2 に示す．総部品数 59 個のうち，4 つの手法で類似度の計算方法が変わっても所属部品群が変わらない部品は 45 個で，いずれの手法でも分析対象外となるのはなかった．残りの 14 個は計算方法によって結果が変わった．3 個はいずれの場合も類似性を確認できず，11 個はいずれかの結果で類似性を確認できた．表 4 のように分類先がどう変化したかを調査したところ，1 個は 1 つの部品群で類似性を確認でき，10 個は複数の部品群で類似性を確認できた．例えば，部品 ID10 の Environment という部品では (内一致度＋外一致度) の時，DefaultEnvironment,HeadlessEnvironment,NonAWTEnvironment と同じ部品群に所属しており，類似性の基準 1,3,4 を満たしている．また，（内一致度＋外一致数），（内一致数＋外一致数）の時，EnvironmentFactory,AbstractOutput が同じ部品群に追加され，部品群が変わったが，基準 1,3,4 を満たしている．

表 1 jlgui での各手法で抽出した部品群と部品数

| 分類手法 | 部品群 | 部品数 | 非分類部品 |
|---|---|---|---|
| (内一致度＋外一致度) | 17 | 58 | 12 |
| (内一致度＋外一致数) | 16 | 55 | 15 |
| (内一致数＋外一致度) | 19 | 66 | 4 |
| (内一致数＋外一致数) | 13 | 58 | 12 |

表 2 barbecue での各手法で抽出した部品群と部品数

| 分類手法 | 部品群 | 部品数 | 非分類部品 |
|---|---|---|---|
| (内一致度＋外一致度) | 12 | 54 | 5 |
| (内一致度＋外一致数) | 13 | 55 | 4 |
| (内一致数＋外一致度) | 11 | 54 | 5 |
| (内一致数＋外一致数) | 12 | 53 | 6 |

## 5 考察

### 5.1 実験のまとめ

実験における観点 1 と観点 2 に関しては，プロジェクト毎，手法毎，類似部品群に含まれる部品の総数，部品群 1 つあたりの部品数がまちまちで，特別な傾向を確認することができなかった．観点 3 として得られた部品群中身を調査した．その結果部品を大きく以下の 5 種類に分けることができた．部品が分類された結果を表 5 に示す．

1. 利用関係がなく，分類の対象外となる．
2. どう分類しても，類似部品群を形成しない．
3. どの分類でも，同じ部品と部品群を形成する．
4. 計算方法によって，類似性を示さなかったり，1 種類の部品群を形成できた．
5. 計算方法によって，複数の部品群を形成する．

表 5 部品の分類結果

| 総部品 | jlgui | barbecue |
|---|---|---|
| 利用関係なし | 4 | 0 |
| 部品群にならず | 8 | 3 |
| 同じ部品群 | 45 | 45 |
| 0 か 1 通り | 7 | 1 |
| 2 通り以上 | 6 | 10 |
| 総ファイル数 | 70 | 59 |

表 3 いずれかの手法で類似性を示しながら異なる分類結果が得られた部品について (jlgui)

| | | 内一致度＋外一致度 | 基準 | 内一致度＋外一致数 | 基準 | 内一致数＋外一致度 | 基準 | 内一致数＋外一致数 | 基準 |
|---|---|---|---|---|---|---|---|---|---|
| ID | クラス名 | 部品群 | 基準 | 部品群 | 基準 | 部品群 | 基準 | 部品群 | 基準 |
| 14 | AbsoluteConstraints | × | × | × | × | 15 | 1,2,3,4 | 15 | 1,2,3,4 |
| 17 | ActiveJBar | 15,70 | × | 15,70 | × | 22 | 1,2,3,4 | 5,18,19,20,22,23,24 | × |
| 20 | ActiveJLabel | 18,19,24 | 1,2,3,4 | 18,19,24 | 1,2,3,4 | 19,21 | 1,2,3,4 | 5,17,18,19,22,23,24 | × |
| 21 | ActiveJNumberLabel | × | × | 25,28 | 2,3 | 19,20 | 1,2,3,4 | × | × |
| 22 | ActiveJPopup | 18,19,24 | 1,2,3,4 | 18,19,24 | 1,2,3,4 | 17 | 1,2,3,4 | 5,18,19,20,23,24 | × |
| 23 | ActiveJSlider | 18,19,24 | 1,2,3,4 | 18,19,24 | 1,2,3,4 | 17,22 | 1,2,3,4 | 5,17,18,19,20,22,24 | × |
| 29 | PlaylistUIDelegate | 48 | × | 10 | 1,4 | 5,25,28,33 | × | × | × |
| 33 | Taftb | 25,28 | 2,3 | 16,49 | × | 5,25,28,29 | × | 25,28 | 2,3 |
| 47 | TagInfoDialog | 34 | 1,4 | × | × | 34 | 1,4 | 36,37,38,39,40 | 1,4 |
| 49 | BMPLoader | × | × | 16,33 | × | 32 | 1,4 | × | × |
| 55 | Array | 12,54 | × | 54,58 | 2,3 | 12,54 | × | 54,58 | 2,3 |
| 58 | SortedStrings | × | × | 54,55 | 2,3 | × | × | 54,55 | 2,3 |
| 63 | PreferenceItem | 60,62,66,68 | 1,2,3,4 | × | × | 60,62,68 | 1,2,3,4 | × | × |

表 4 いずれかの手法で類似性を示しながら異なる分類結果が得られた部品について (barbecue)

| | | 内一致度＋外一致度 | 基準 | 内一致度＋外一致数 | 基準 | 内一致数＋外一致度 | 基準 | 内一致数＋外一致数 | 基準 |
|---|---|---|---|---|---|---|---|---|---|
| ID | クラス名 | 部品群 | 基準 | 部品群 | 基準 | 部品群 | 基準 | 部品群 | 基準 |
| 3 | BarcodeFactory | 14,19,50,52,58 | × | 27 | 1,4 | 7,27,42,52,56,59 | 1,4 | 27 | 1,4 |
| 8 | CompositeModule | 7,42,50,52,58 | 1,4 | 7,42,,56,59 | 1,3,4 | 17,21,25,30,33,35,37 | 1,4 | 1,28,31,40,57 | 1,4 |
| 10 | Environment | 9,12,13 | 1,3,4 | 9,11,12,13,44 | 1,3,4 | 9,12,13,44 | 1,3,4 | 9,11,12,13,44 | 1,3,4 |
| 11 | EnvironmentFactory | × | × | 9,10,12,13,44 | 1,3,4 | × | × | 9,10,12,13,44 | 1,3,4 |
| 20 | CharBuffer | 17,21,25,30,33,35,37 | 1,3,4 | × | × | × | × | × | × |
| 27 | BooklandBarcode | × | × | 3 | 1,4 | 3,7,42,52,56,59 | 1,4 | 3 | 1,4 |
| 31 | LinearBarcode | 28,40 | 1,3,4 | 28,40 | 1,3,4 | 1,28,40,57 | 1,4 | 1,8,28,40,57 | 1,4 |
| 42 | Module | 7,8,56,59 | 1,3,4 | 7,8,56,59 | 1,3,4 | 3,7,27,52,56,59 | 1,4 | × | × |
| 44 | AbstractOutput | × | × | 9,10,11,12,13 | 1,3,4 | 9,10,12,13 | 1,3,4 | 9,10,11,12,13 | 1,3,4 |
| 50 | LabelLayoutFactory | 14,19,52,58 | × | 45,46,51 | 1,3,4 | 14,19,49,58 | × | 45,46,51 | × |
| 52 | Output | 3,14,19,50,58 | × | 47,48,54,55 | 1,3,4 | 3,7,27,42,56,59 | 1,4 | × | × |

中でも，計算方法によって複数の部品群を形成しうる部品は，複数の観点から他の部品との類似性を見出していると考えられ，jlgui の場合は全体 1 割，barbecue の場合は 2 割の部品が該当した．これらの部品の存在を考えると，4 つの方法でそれぞれ類似度を計算し，それぞれの結果を比較しながら判定することが望ましいと考えた．

### 5.2 部品理解での活用について

コードブラウザ上などで，類似部品を紹介する際に 4 つの方法それぞれで類似度を計算する方法を用いることで，前述での分類が，活用できると考えられる．例えば，どの分類でも同じ部品と部品群を形成する場合強い類似性を有すると紹介することができたり，複数の観点で類似性を示している場合，それぞれの観点を紹介することでより多面的な説明ができると考えられる．このような形で結果を組み合わせることでの精度向上を期待したい．

## 6 まとめ

本研究では，ソフトウェア内外の部品の利用状況に基づいて部品の分類手法を，一致度と類似部品数による類似度計算の組み合わせから 4 通り提案し，分類結果の違いを調査した．4 通り全てで同じ分類結果になる部品がほとんどである一方で 1〜2 割の部品は複数の類似部品群に属する結果になり，それぞれの結果を組み合わせることが有効ではないかと考えた．今後他のソフトウェアに対しても適用することで，同様の結果を得られるかを調査したい．

## 参考文献


[1] Zhong，T.Xie，L.Zhang，J.Pei，and H.Mei，” Mapo:Mining and recommending apiusage patterns，” in proceedings of the 23rd European Conference on Object-Oriented Programming(ECOOP2009)，2009，pp.318-343.

[2] Reishi Yokomori, Norihiro, Yoshida, Masami, Noro, Katusro, Inoue,:”Use-Relationship Based Classification for Software Components”, Proceedings of the 6th International Workshop on Quantitative Approaches to Software Quality (QuASoQ 2018), pp.59-66,2018.

[3] 藤田翔大，清水太智，戸本了太：“利用部品の共通性に基づくソフトウェア分類手法類似度計算手法に関する考察”，南山大学理工学部 2019 年度卒業論文，2020.

[4] 青山季暉，名和大騎：“利用関係の一致度に基づくソフトウェア部品分類手法ソフトウェア外で定義された部品の利用関係を加味した手法の実現”，南山大学理工学部 2020 年度卒業論文，2021.