特別研究報告

題目

設計情報の再利用を目的としたＵＭＬ図の自動推薦ツール

指導教員
井上 克郎 教授

報告者
松下　誠

平成 22 年 2 月 16 日

大阪大学 基礎工学部 情報科学科

平成 21 年度 特別研究報告

設計情報の再利用を目的とした UML 図の自動推薦ツール

松下　誠

内容梗概

ソフトウェア開発において，適切な再利用を行うことによりソフトウェアの生産性や信頼性が向上するとされている．なぜなら，新規に作成する場合と，以前作成されたものを利用する場合では製作にかかる時間が大きく異なり，また，新規に作成したものと比べると，テストなどを重ねた既存の成果物の信頼性は高いからである．再利用の対象は仕様書や設計図などの設計情報と，クラスやメソッドといったソースコードなどのプログラムの二つに大きく分けることができる．

現在，過去に開発された設計文書を再利用する手法としては，設計図間の類似性を計測する手法や，設計図の類似性を利用して設計図を検索する手法が提案されている．しかし，設計情報の再利用は，ソースコードなどの再利用に比べてツールによるサポートが立ち遅れているといわれている．

ところで，再利用を支援する手法では，開発者が検索に利用するキーワードを決定する検索手法が採られていることが多い．しかし，このようなキーワードによる検索には問題点が二つある．まず一つは，開発者が再利用可能な情報の存在に気付かなければ検索が行われないという点である．そしてもう一つは，開発者が適切なキーワードを選ばなければならないという点である．これらのことから，再利用の機会を逸しないために，検索はシステムが能動的に行う必要がある．

以上のことから，本研究では開発者が検索の実行を指示せずとも，自動的に再利用可能な UML 図の検索を行う自動推薦ツールを作成した．本ツールでは，作成中の UML 図に現れるテキスト情報を取得し，それと類似した情報を持つ UML 図の検索を自動的に行う．

評価実験として，本ツールが UML 図の作成中に適切な類似 UML 図の推薦を行うことができることを確認する実験を行った．その結果，開発者が UML 図の一部分を記述すれば，適切な類似 UML 図が推薦されることが分かった．

主な用語

UML 図
再利用
自動推薦

目 次

1　まえがき

ソフトウェア開発の上流工程で問題を見逃がしてしまい下流工程で問題を取り除く必要が生じた場合，修正に必要なコストが増大することが知られている．一方，ソフトウェア再利用を適切に行うことによって，ソフトウェア開発の生産性と成果物の信頼性の両方が向上すると言われている [1]．ソフトウェア再利用の対象としては，ソースコードなどの実行可能なプログラムと，設計情報を記録した文書の 2 つがある．そのため，再利用によって上流工程の成果物の品質を改善することは，開発コストの低減に有効な手段であると考えられる．

設計情報を積極的に複数のプロジェクトで共有する開発手法に，プロダクトライン開発 [2] がある．プロダクトライン開発は同じ種類のソフトウェア製品を系統的に開発する手法であり，製品間で共通する部分を，共通の設計図を用いて開発する．しかし，プロダクトライン開発では，過去に開発したソフトウェア製品の設計情報を再利用することは考慮されていない．

過去に開発された設計文書を再利用するためには，設計図間の類似性を計測する手法 [3, 4] や，設計図の類似性を利用して設計図を検索する手法 [5] がある．しかし，設計情報の再利用は，ソースコードなどの再利用に比べてツールによるサポートが立ち後れていると言われている [6].

一方，ソースコードの再利用は設計文書に比べて活発に研究されており，ソースコードを検索するシステム [7, 8, 9] などの，再利用を支援する様々な手法が提案されている．このような再利用を支援する手法の一種に，再利用に適していると考えられるソフトウェア部品を，ソフトウェア開発中に自動的に推薦する手法 (ソフトウェア部品の自動推薦) がある [10, 11]. 自動推薦を行う利点としては，ソフトウェア開発者が再利用可能な情報の存在に気付いていない場合であっても，再利用を行うチャンスを提供できることが挙げられる．

そこで，本研究では，蓄積した設計情報の再利用を促進することを目的として，設計情報の自動推薦ツールを提案する．提案手法は，開発者が UML 図を編集している時に，編集中の UML 図と関連があると考えられる UML 図を提示するツールである．提案するツールにより，設計情報の柔軟な再利用が行なわれると期待される．

以下，2 章では提案手法である UML 図の自動推薦手法について述べる．3 章ではツールの実装の概要を述べ，4 章でツールを用いた実験について説明する．最後に，5 章で本研究のまとめと課題を述べる．

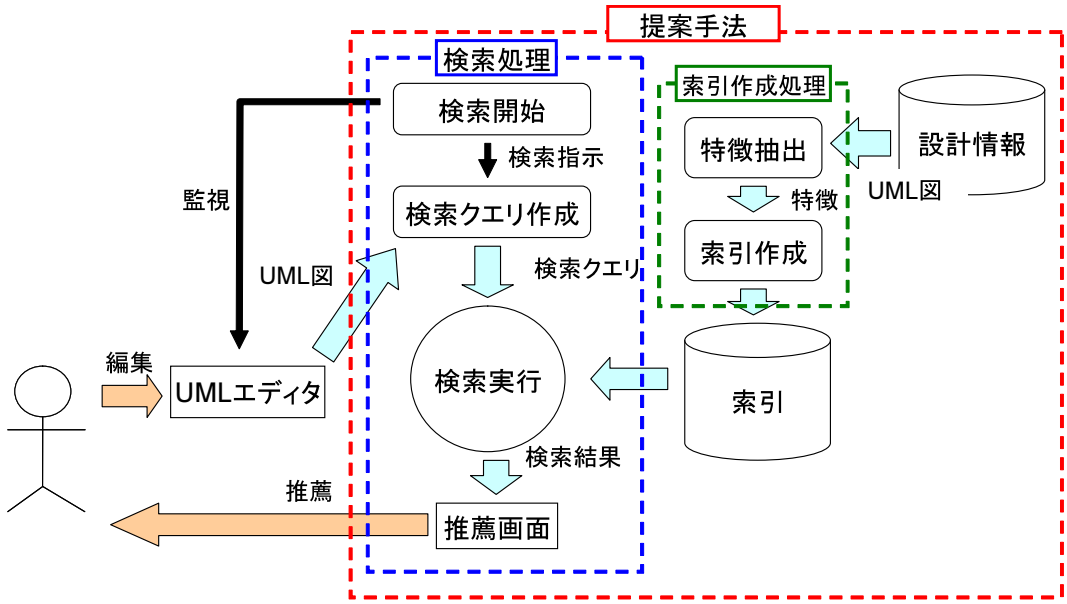


図 1: 提案手法の概要

2 設計情報の自動推薦手法

ここでは開発者が UML 図を作成しているときに，過去の設計情報を自動的に推薦する手法について述べる．具体的には，UML 図の作成中に現在の UML 図と類似した過去の設計情報を自動的に検索し，開発者に提示する．開発者は，推薦された UML 図を確認し，再利用したいものがあれば現在作成している UML 図へ取り込む．検索を自動的に行うことで開発者が意図せずとも過去の設計情報が提示され，再利用の機会が増加することが期待できる．

図 1 に本手法の概要を示す．本手法では，開発者の作成する UML 図を入力として，索引を用いた検索により過去の設計情報から入力と類似した設計情報を出力する．本手法の内部処理は大きく索引作成処理と検索処理の二つに分けることができる．

まず，過去の設計情報から特徴と呼ぶ情報を抽出し，それを基に検索に必要な索引を作成する．ここでいう特徴とは，UML 図中の各ノードのテキスト記述を加工することで得られる単語である．

次に，開発者の編集状況に応じて作成中の UML 図から同様に特徴を抽出し，索引を用いて類似 UML 図を検索する．そして，検索結果を再利用できる可能性が高い設計情報として，開発者に提示する．

本手法を用いて UML 図の設計を行うことで過去の設計情報から類似した設計情報を検索する手間が省け，容易に再利用を行うことができるようになる．

なお，本手法の検索には潜在的意味インデキシング (Latent Semantic Indexing) [12] を用

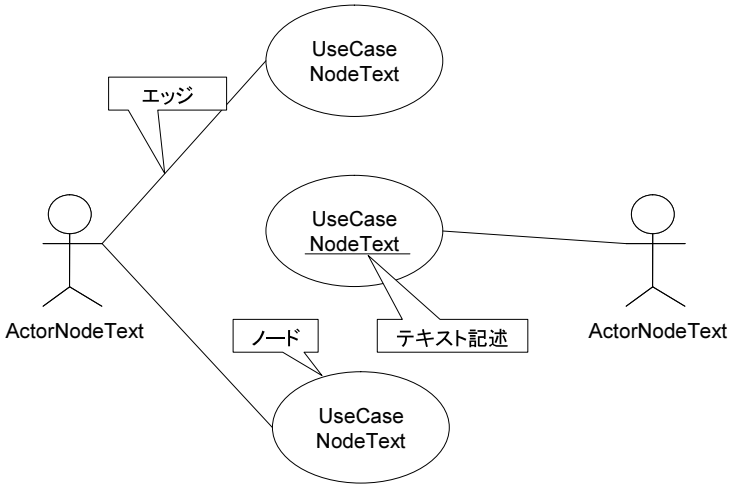


図 2: UML 図のデータ構造 (ユースケース図)

いる．潜在的意味インデキシングでは文章とそれに含まれる単語の出現回数を共起行列として表現するが，本手法では文章に UML 図を，単語に特徴を対応させる．これを用いることにより，曖昧さを許容した検索が行われることが期待できる．また，本手法の索引作成手法，及び検索手法はソフトウェア部品の自動推薦システムである A–SCORE [11] に基づいている．

2.1 UML 図

UML(unified modeling language) は，ソフトウェアシステムをモデル化するための図のかき方であり，UML 図は UML に従ってかかれた図である．

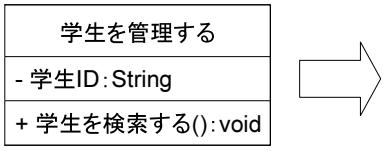
次に，図 2 に UML 図の一つであるユースケース図の例を示し，本手法で用いる UML 図のデータ構造について説明する．UML 図はノードとエッジで構成されたグラフ構造を持つ．各ノードはラベルとしてテキスト記述を持っており，本手法ではテキスト記述を利用する．

2.2 特徴の抽出

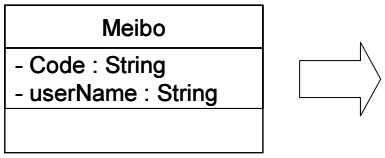
本手法では，設計情報の UML 図を解析して特徴を抽出する．本節では UML 図から特徴を抽出する方法を説明する．

本手法で利用する特徴とは，前述したように UML 図中の各ノードのテキスト記述を加工することで得られる単語である．そして，ノードのテキスト記述を以下の手順により得られた単語が特徴である．

1. ノードのテキスト記述をスペースと，ピリオド・カンマ・アンダーバーといった記号



特徴	出現回数
学生	3
管理する	1
検索する	1
学生を管理する	1
学生を検索する	1
id	1
string	1
void	1



特徴	出現回数
meibo	1
code	1
string	2
user	1
name	1

図 3: 特徴の抽出例

類で区切り，文字列に分割する．

2. 1で得た文字列を CamelCase[1]に従っているものとみなし，小文字に続く大文字の直前で分割する．

3. 2で得た文字列が日本語と英単語の複合語の場合，日本語と英単語の間で分割する．

4. 3で得た日本語の文字列を助詞で区切り，分割する．

5. 大文字を小文字に変換する．

6. 検索対象となる過去の設計情報の半数以上の UML 図に現れる単語は検索に有用でないと判断し，除去する．

ただし．大文字が連続している部分は頭字語であるので一つの単語とする．また，助詞と判定される文字が名詞の一部に入っている場合など，日本語の単語を助詞で区切るのは適切でない場合があるので，日本語の単語に関しては助詞で分割する前と後の両方を特徴として扱う．

図 3 に，特徴抽出の例を示す．まず，上の例では,「学生を管理する」と「学生を検索する」は，助詞の「を」で分割され,「学生」と「管理する」,「検索する」の三種類の特徴となって

[1]CamelCase とは，複数の単語からなる語を表記する際に，各単語の先頭の文字を大文字で表記するという手法である．

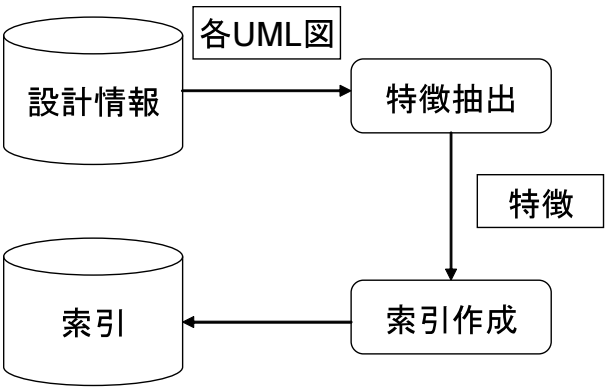


図 4: 索引作成の流れ

いる．そして，日本語の単語は助詞で分割する前と後の両方を特徴として扱うというルールによって，分割する前の「学生を管理する」と「学生を検索する」の二つも特徴となっている．さらに,「学生 ID」は日本語と英単語の複合語であるので,「学生」と「ID」の二語に分割されている．

次に，下の例では「userName」が CamelCase と判断されるため,「user」と「Name」の二語に分割されている．

最後に，両方の例において，各単語はすべて小文字に変換されている．

2.3 索引の作成

抽出した特徴から検索に用いる索引を作成する (図 4)．索引は得られた特徴を潜在的意味インデキシングの手法に従って変換したもので，UML 図ファイル名リスト，特徴リスト，索引 UML 図行列，索引特徴行列から成る．

潜在的意味インデキシングでは，UML 図と特徴の関係を潜在的なトピックを用いて UML 図―トピックの関係と，トピック―特徴の関係に分解し，これらを索引とする．索引 UML 図行列は，そのうちの UML 図―トピックの関係を表す行列であり，各 UML 図が各トピックをどの程度含んでいるかを値として持っている．また，索引特徴行列はトピック―特徴の関係を表す行列であり，各特徴が各トピックにどの程度属しているかを値として持っている．

索引の作成は，まず各 UML 図の特徴を元に，列が UML 図に，行が特徴に対応する UML 図―特徴の共起行列 A を作成する．特徴の数を N，UML 図の数を M とすると，A は $N \times M$ 行列となる．行列の ij 要素は j 番目の UML 図における i 番目の特徴の出現回数となる．また，この行列の N 次元列ベクトルを UML 図ベクトルと呼ぶ．

この行列から，潜在的意味インデキシングの手法に従って索引を得る．

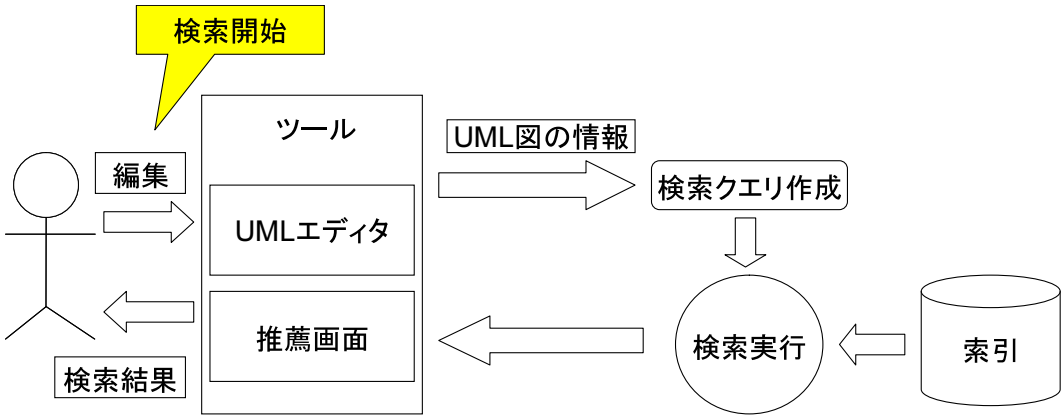


図 5: 検索の流れ

2.4 検索アルゴリズム

検索は，2.3 節で作成した索引を用いて行う．本手法における処理手順を以下と図 5 に示す．

1. あらかじめ定めた検索のタイミングに従って，検索を開始する．
2. 編集中の UML 図から特徴を抽出し，検索クエリを生成する．
3. 生成した検索クエリと索引を用いて類似 UML 図を検索する．
4. 検索結果を開発者に提示する．

以降，各手順について詳細を述べる．

2.4.1 検索開始のタイミング

本手法では，検索処理が自動で行われるため，検索開始のタイミングは検索に用いる特徴の変化する時が望ましい．そこで，検索開始のタイミングを以下の 2 点に定めた．

- ノードのテキスト記述の編集作業が終わった．
- ノードが削除された．

これらのイベントが検出された場合，検索クエリを生成し，検索を行う．

2.4.2 検索クエリの生成

作成中の UML 図から特徴を抽出し，検索条件を決定する検索クエリを生成する．検索クエリは＜特徴，重み＞の組の集合であり，重みは各特徴が作成中の UML 図に表れる回数である．

検索クエリ生成に利用する特徴は，索引作成の特徴の抽出と同様の方法で作成中の UML 図を解析することで取得する．

2.4.3 検索の実行

2.3 節で作成した索引を用いて，検索クエリに合う設計情報を潜在的意味インデキシングの手法に従って検索し，順位付けを行う．

まず，検索クエリを擬似 UML 図ベクトルに変換する．擬似 UML 図ベクトルは，検索クエリに含まれる特長の重みを UML 図ベクトルと同じ順で並べたベクトルである．

次に擬似 UML 図ベクトルに類似した UML 図ベクトルを持つ UML 図を索引を用いて検索し，類似度の高いもの数個を検索結果とする．

2.4.4 推薦の方法

検索アルゴリズムによる検索結果の一覧を開発者に提示する．開発者に提示される情報は，検索結果の順位が 10 位以内である設計情報のファイル名のリストである．

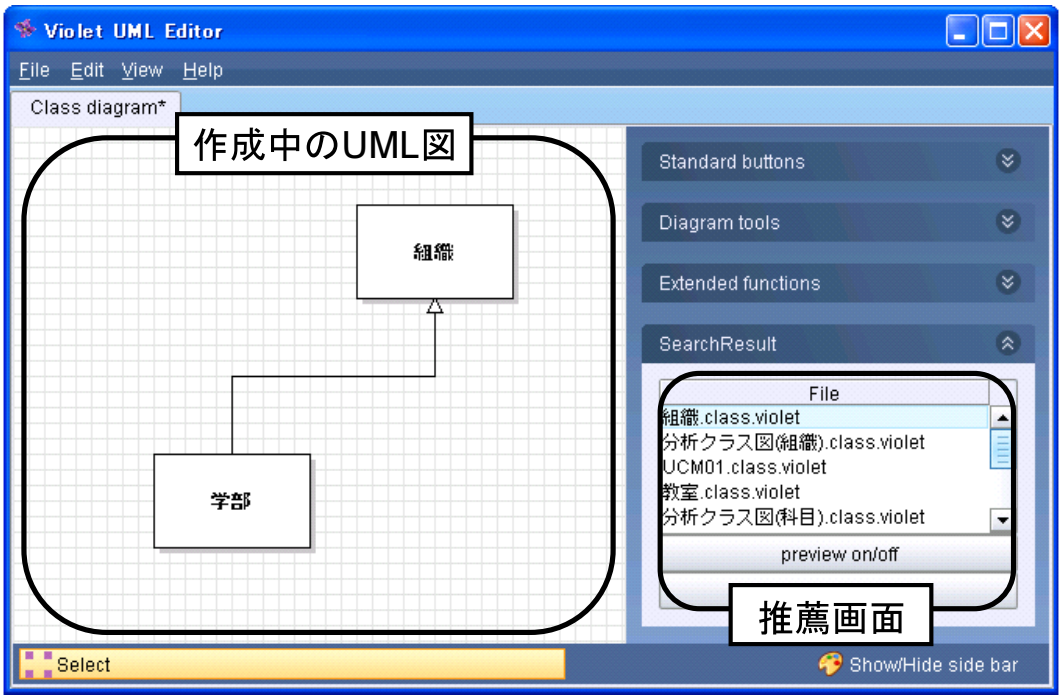


図 6: ツールの動作画面

3　ツールの実装

2 章で述べた UML 図の自動推薦手法を実装したツールを作成した．本章では，本ツールの実装の詳細について説明する．まずツールの外観を図 6 に示す．

3.1　UML エディタ

開発者は本ツールの UML エディタ部分 (図 6 左部) を用いて UML 図の作成と編集を行う．本ツールでは UML エディタとして Violet UML Editor [13] を使用した．Violet UML Editor では 6 種類の UML 図の利用が可能である．以下に，利用可能な UML 図を示す．

- クラス図
- ユースケース図
- アクティビティ図
- シーケンス図
- オブジェクト図

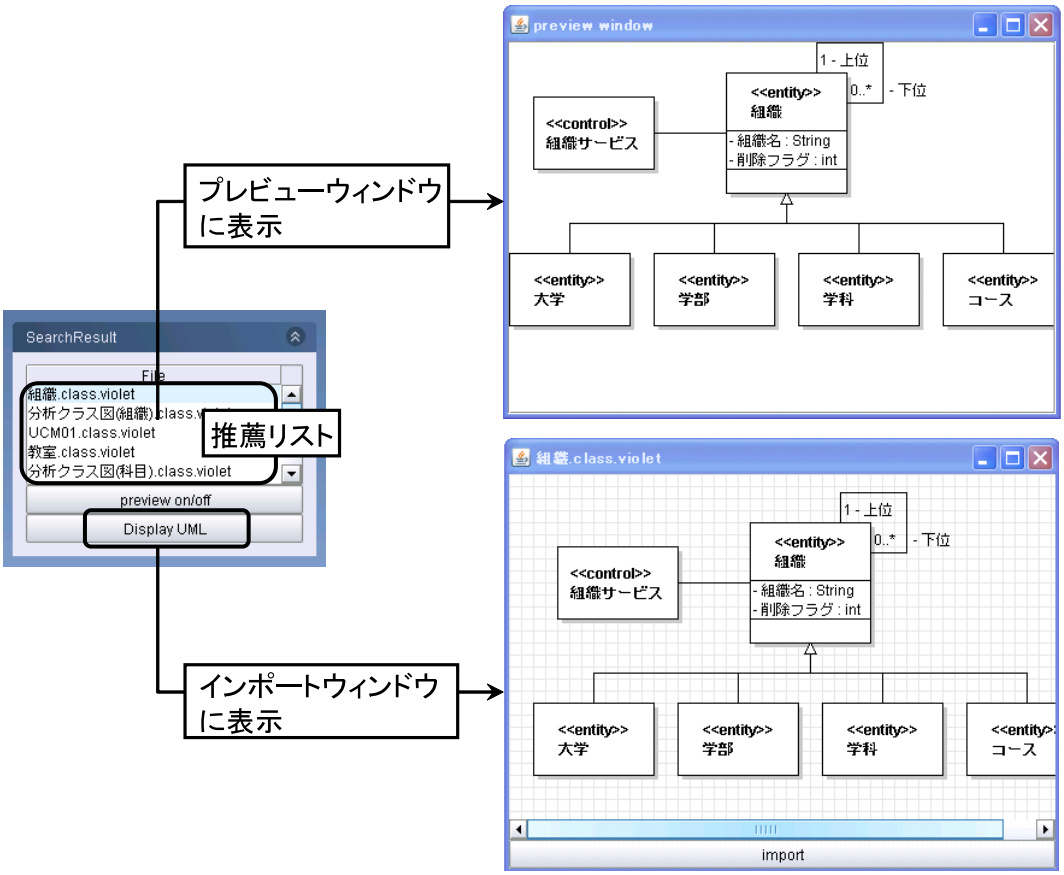


図 7: 推薦画面で選択している UML 図の画像がプレビューウィンドウ (図右上部) に，Display UML ボタンを押すと選択している UML 図がインポートウィンドウ (図右下部) にそれぞれ表示される．

- 状態遷移図

3.2 推薦結果の提示

開発者の UML エディタ部での編集作業に応じて，2.4.1 節で定めたタイミングで自動的に検索が行われ，推薦結果が推薦画面 (図 6 右下部) に出力される．なお，推薦画面に表示される設計情報は，検索順位の上位 10 個である．

3.3 プレビュー機能

この機能は，推薦された UML 図を確認するためのものである．推薦画面に出力された UML 図を選択すると，preview ボタンにより表示・非表示の切り替えが可能なプレビューウィンドウに UML 図が表示される (図 7)．また，検索が行われると，検索順位が最上位の

UML 図が表示される．このため，開発者はツールが最も類似度が高いと判断した UML 図を確認しながら UML 図の作成を行うことができる．

なお，プレビュー機能では UML 図を画像ファイルとして出力したものを利用する．あらかじめ画像ファイルを用意しておくことで，推薦された UML 図のプレビューを高速で見ることができる．この画像ファイルは本ツールで作成可能である．

3.4 インポート機能

この機能は，推薦された UML 図を現在編集中の UML エディタ部へ取り込むためのものである．

推薦画面に提示された設計情報を選択し，Display UML ボタンを押すと，選択した UML 図の取り込みを行うためのインポートウィンドウが表示される (図 7)．

インポートウィンドウ上で再利用したい範囲を選択し，import ボタンを押すことで UML 図の選択範囲が取り込まれる．UML 図の取り込み先は，取り込もうとした UML 図と現在編集中の UML 図の種類が同じならば，現在のエディタ部へ取り込まれ (図 8)，取り込もうとした UML 図と現在編集中の UML 図の種類が異なるならば，新規のエディタ部を開き，そこに取り込む (図 9)．この時，再利用したい範囲の選択を行っていなかった場合は，選択した UML 図全体が取り込まれる．

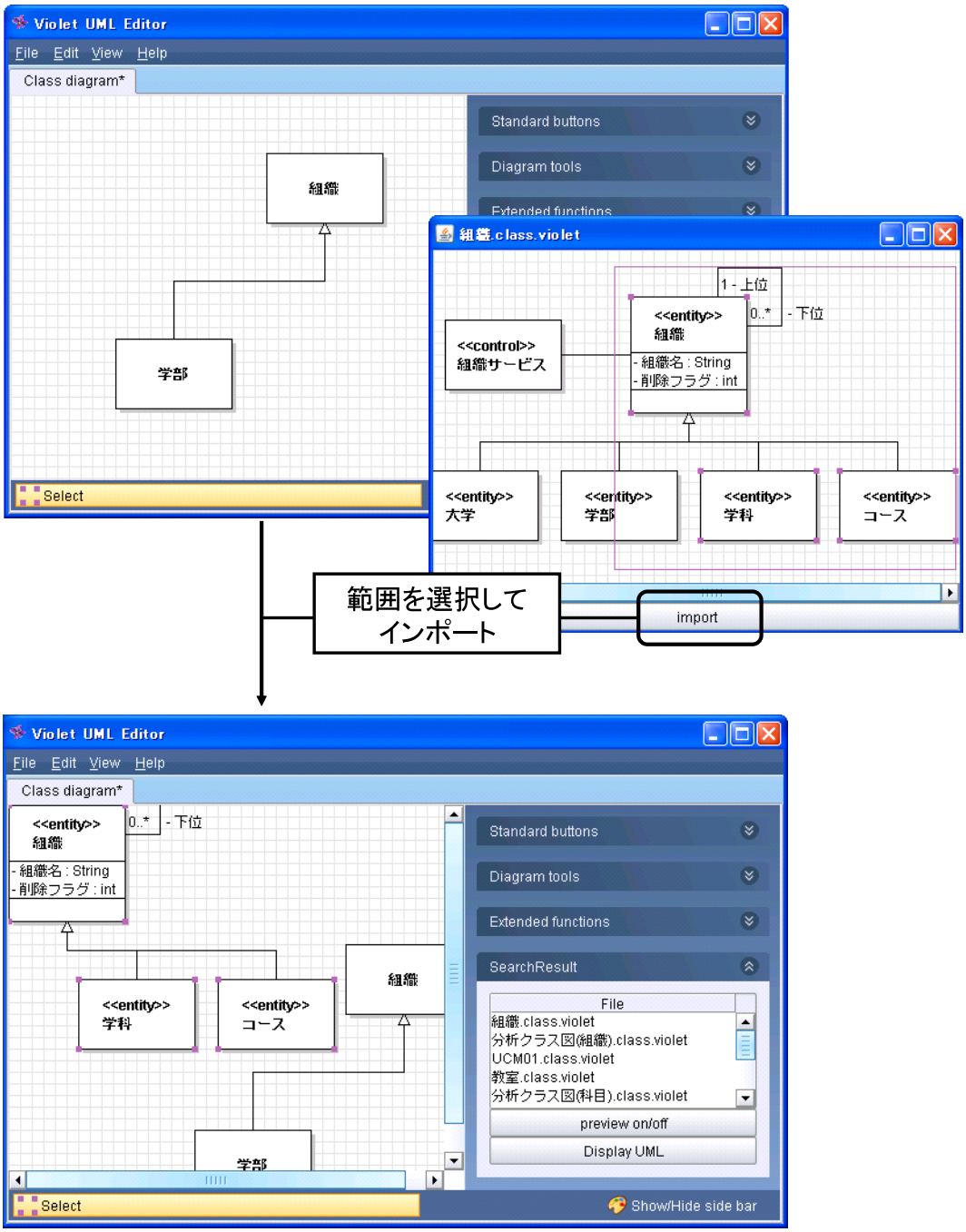


図 8: インポート範囲を選択して import ボタンを押すと，UML 図がインポートされる．

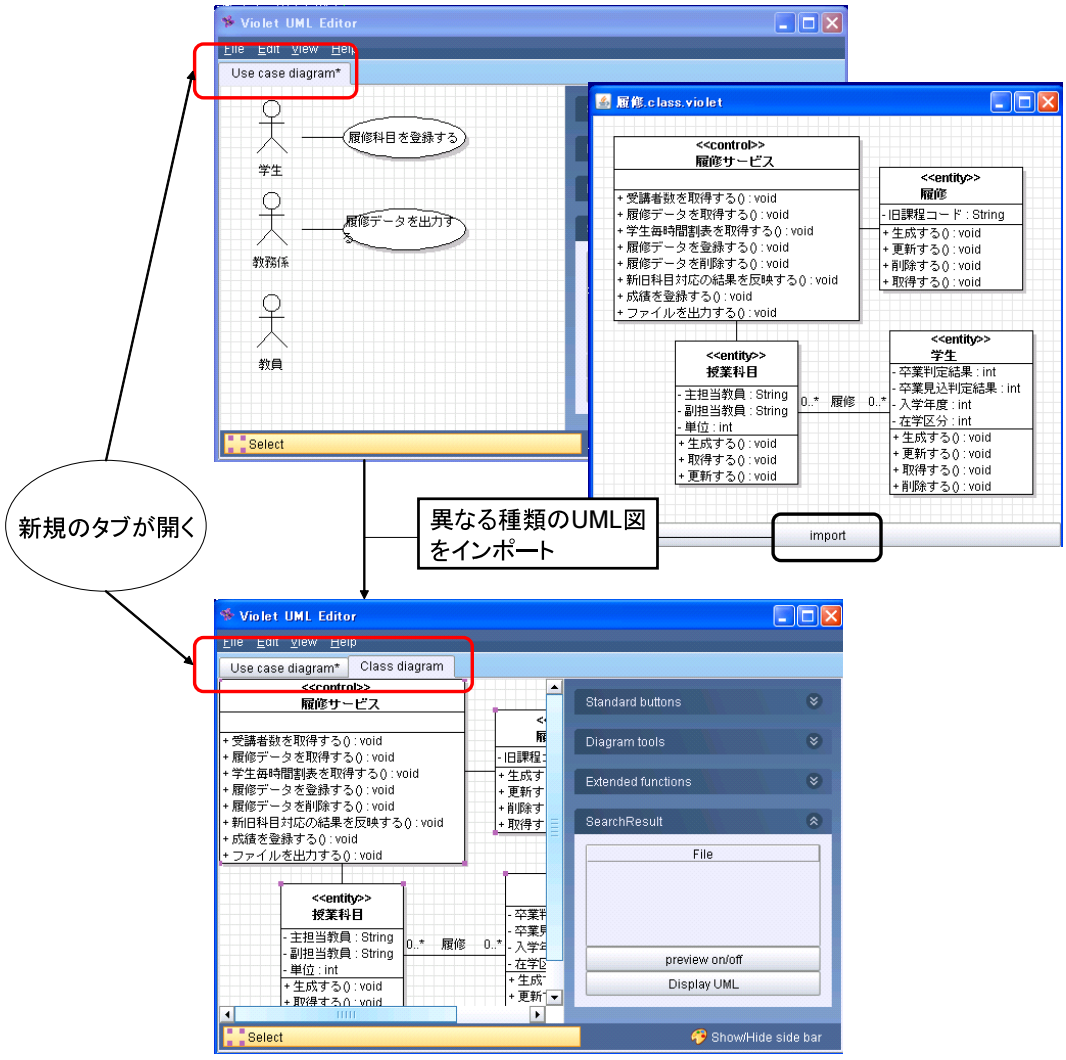


図 9: 編集中の図と異なる種類の UML 図のインポートを行うと，新規のタブが開かれ，そこにインポートされる．

4　評価実験

本ツールの評価実験を行った．

本実験では推薦されるべき UML 図が存在する場合，その UML 図の一部を記述することで元の UML 図が推薦されることを確認するために，UML 図の持つ特徴の一部を用いて推薦処理を行った．そしてその結果，有効な推薦が行われることを確認した．

4.1　実験の目的

本実験は，開発者が本ツールを用いて UML 図の設計を行った場合，どの程度の作図を行えばデータベースから適切な類似 UML 図が推薦されるのか，ということを確認するために行った．

4.2　実験の内容

まず UML 図のリポジトリを作成した．次に，リポジトリから UML 図を一つ選び，その UML 図の持つ特徴情報の一部をランダムに取り出して検索クエリを生成し，検索を行った．この時，元の UML 図が推薦順位上位に推薦されていれば，有効な検索が行われたと判断する．そして，UML 図とランダムに取り出す特徴の割合を変化させながら繰り返し試行を行うことで，取り出す特徴の割合と UML 図の推薦順位の関連性の調査を行った．

この試行の各一回は，開発者が UML 図の一部を記述した際に行われる推薦と同じである．つまり，この試行において有効な検索が行われれば，開発者が実際に本ツールを用いて UML 図の設計を行った場合にも有効な検索が行われると言うことが出来る．

なお，ランダムに取り出す特徴の割合は，検索対象となる UML 図の持つ特徴数の 5%から 100%まで 5%刻みとした．さらに，ランダムに取り出す特徴の偏りなどによる実験結果の偏りを防ぐため，各試行はそれぞれ 10 回ずつ行った．

4.3　リポジトリの構築

推薦対象となる設計情報のリポジトリの入力データには，実プロジェクト教材 [14] の一部である，UML で記述された大学の履修管理システムの設計情報を用いた．この入力データは全部で 91 個あり，ユースケース図とクラス図，シーケンス図で構成されている．各 UML 図の個数の内訳は，ユースケース図が 10 個とクラス図が 73 個，シーケンス図が 8 個である．

そして，入力に対する索引の作成を行った．その結果得られた索引の各パラメータを以下に示す．

- 特徴の種類数：469 語

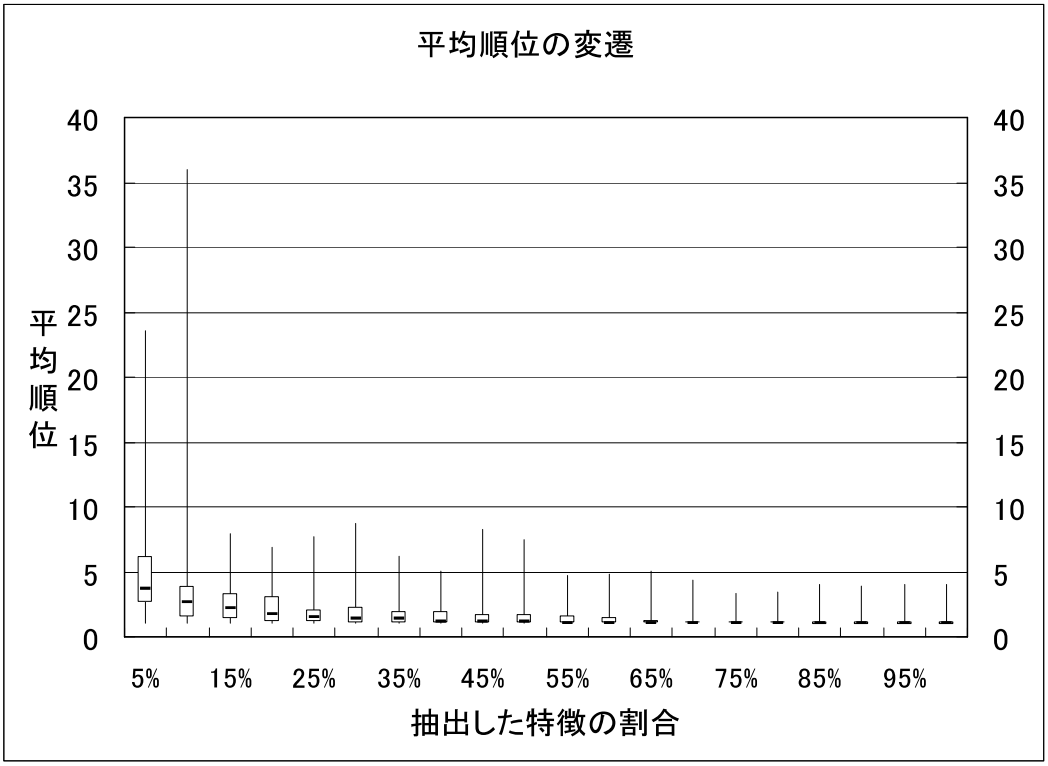


図 10: 平均順位の変遷

- LSI による次元縮約のパラメータ：40
- 索引のサイズ：176KB

4.4 実験の結果

図 10 は，抽出する特徴の割合の変化による元の UML 図の平均推薦順位の変遷を表す箱ひげ図である．X 軸と Y 軸に，それぞれ抽出した特長の割合と平均順位を取る．また，長方形の下側が平均順位の分布の第 1 四分位を表し，上側が第 3 四分位を表し，中央線が中央値を表している．

抽出する特徴の割合が 100%に近づくほど，平均推薦順位が高順位になっている．抽出する特徴の割合が 10%以下の場合は推薦順位が低いこともあるが，それより高い値では，リポジトリ内のすべての図で推薦順位が上位 10 位以内となっている．

このことから，開発者が元の UML 図の 10%程度の作成を行えば，目的の UML 図が推薦されることがわかった．

図 11 は，抽出する特徴の割合による元の UML 図が推薦順位上位に入っている割合を示

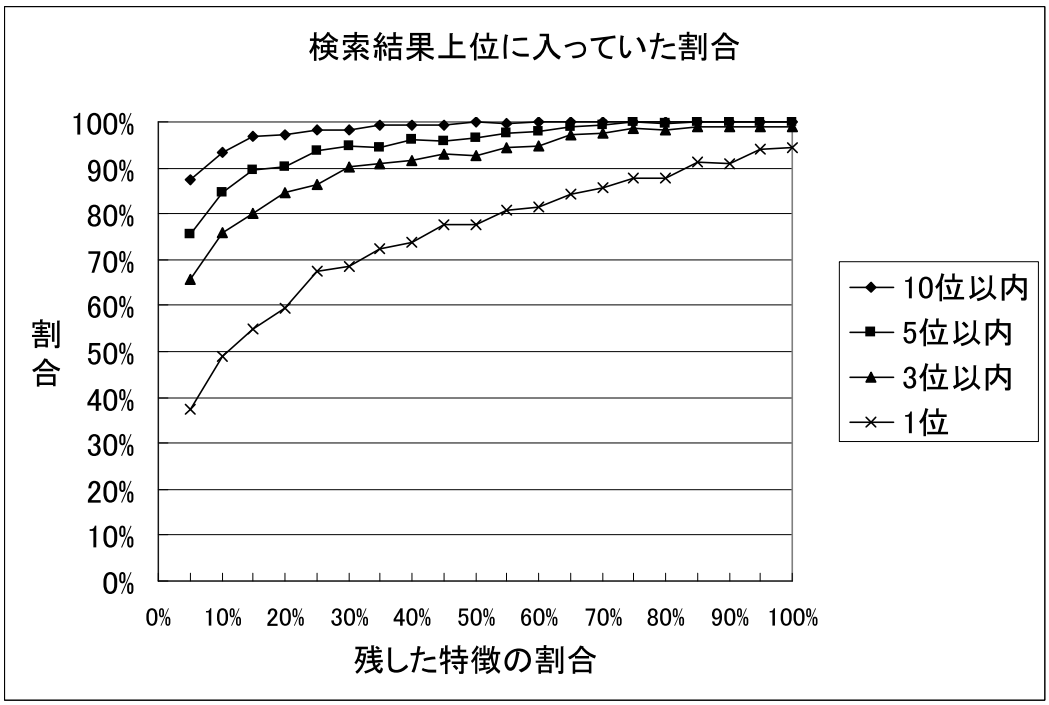


図 11: 検索結果上位に入っていた割合

す折れ線グラフである．X 軸と Y 軸に，それぞれ抽出した特長の割合とその時に元の UML 図が一定以上の順位に推薦されていた割合を取る．4 つの折れ線はそれぞれ推薦順位が 10 位以内のとき，5 位以内のとき，3 位以内のとき，1 位のときの割合を表している．

抽出する特徴の割合が 100%に近づくにつれて，推薦順位が上位となる割合も上がっている．特に，推薦順位が 1 位となる場合については，特徴の割合が 10%のときに 50%，50%のときには 80%となっている．

このことから，作図作業の早い段階でも，目的の UML 図が高い順位で推薦されることがわかった．

4.5 考察

本実験では，本ツールが適切な推薦を行うことができるかを調査するために，リポジトリを利用した推薦を行った．実験結果から，抽出する特徴の割合が低い段階でも上位に推薦されることが多く，さらに，抽出する特徴の割合が増えるにつれて元の UML 図の推薦順位が向上することがわかった．このことから，リポジトリに推薦図がある場合の推薦精度は高いと考えられる．

次に，実験結果の妥当性について述べる．

まず，実験で用いた設計情報が持つ特徴の一部をランダムに抽出する方法であるが，これは計算機によって生成される乱数を用いて行った．さらに，乱数の偏りが実験に与える影響を軽減するため，各試行を 10 回ずつ行った．これにより，実験結果の信頼性が向上する．

また，今回リポジトリの設計情報の約 8 割がクラス図であった．クラス図以外の UML 図に対する実験結果が，クラス図に対する実験結果と著しく異なる場合，実験結果の信頼性は低下する．しかし，検索に利用している特徴はテキスト情報であり，UML 図による違いはない．さらに，リポジトリ内にあったユースケース図やシーケンス図でも，クラス図と同様の精度で推薦結果が出ている．よって，扱う UML 図の種類がクラス図に偏っていることによって実験結果の信頼性が低下することは少ないと考えられる．

次に，実験の一般性について述べる．実際にツールを使用する場合，リポジトリに推薦の対象となる設計情報が入っていないということが考えられる．その場合に推薦される設計情報は開発者にとって有用ではなく，設計情報が提示されることで作業効率の低下を招く可能性がある．今後は被験者を用意して，推薦が作業効率に与える影響などを計測する実験を行う必要があると考えられる．

5　まとめ

本研究では，設計情報の再利用に着目し，編集中の UML 図に類似した UML 図を開発者に提示する自動推薦ツールを作成した．本ツールでは，UML 図中のノードのテキスト記述を利用し，潜在的意味インデキシングを用いた曖昧さを許容した検索を行う．

そして，本ツールの検索精度を確認するためにリポジトリの UML 図を推薦する評価実験を行った．

今後の課題としては，推薦結果の有用度を向上させるために，ノードのテキスト記述だけでなく，検索に利用する情報として UML 図のグラフ構造も扱うようにしたいと考えている．また，リポジトリに無い図を作成する場合の推薦結果が開発者の作業効率に与える影響を調べるために，被験者を用いた実験が必要である．

謝辞

本研究において，常に適切な御指導および御助言を賜りました大阪大学大学院情報科学研究科コンピュータサイエンス専攻 井上 克郎 教授に心より深く感謝いたします.

本研究において，適切な御指導および御助言を賜りました大阪大学大学院情報科学研究科コンピュータサイエンス専攻 松下 誠 准教授に深く感謝いたします.

本研究において，逐次適切な御指導および御助言を頂きました大阪大学大学院情報科学研究科コンピュータサイエンス専攻 石尾 隆 助教に深く感謝いたします.

本研究において，様々な御指導および御助言を頂きました大阪大学大学院情報科学研究科コンピュータサイエンス専攻 早瀬 康裕 特任助教に深く感謝いたします.

最後に，その他様々な御指導，御助言等を頂いた大阪大学大学院情報科学研究科コンピュータサイエンス専攻井上研究室の皆様に深く感謝いたします.

参考文献

[1] Charles W. Krueger. Software reuse. *ACM Comput. Surv.*, Vol. 24, No. 2, pp. 131–183, 1992.

[2] John J. Marciniak. *Encyclopedia of Software Engineering*, pp. 433–434. John Wiley & Sons, Inc., 2002.

[3] Motoshi Saeki. Patterns and aspects for use cases: Reuse techniques for use case descriptions. In *Proceedings of the 4th International Conference on Requirements Engineering (ICRE'00)*, pp. 62–, Washington, DC, USA, 2000. IEEE Computer Society.

[4] Maurits C. Blok and Jacob L. Cybulski. Reusing uml specifications in a constrained application domain. *Software Engineering Conference, 1998. Proceedings. 1998 Asia Pacific*, pp. 196–202, 1998.

[5] William N. Robinson and Han G. Woo. Finding reusable uml sequence diagrams automatically. *IEEE Software*, Vol. 21, No. 5, pp. 60–67, 2004.

[6] Axel van Lamsweerde. Requirements engineering in the year 00: a research perspective. In *International Conference on Software Engineering*, pp. 5–19, 2000.

[7] Katsuro Inoue, Reishi Yokomori, Tetsuo Yamamoto, Makoto Matsushita, and Shinji Kusumoto. Ranking significance of software components based on use relations. *IEEE Trans. Software Engineering*, Vol. 31, No. 3, pp. 213–225, 2005.

[8] Google Code Search. `http://www.google.com/codesearch`.

[9] Koder. `http://www.koders.com`.

[10] Yunwen Ye and Gerhard Fischer. Reuse-conducive development environments. *Automated Software Engg.*, Vol. 12, pp. 199–235, April 2005.

[11] 島田次, 市井誠, 早瀬康裕, 松下誠, 井上克郎. 開発中のソースコードに基づくソフトウェア部品の自動推薦システム A–SCORE. 情処学論, Vol. 50, No. 12, pp. 3095–3107, December 2009.

[12] Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. Indexing by latent semantic analysis. *J. Am. Soc. Inf. Sci.*, Vol. 41, No. 6, pp. 391–407, 1990.

[13] Violet UML Editor. http://alexdp.free.fr/violetumleditor/page.php.

[14] 実プロジェクト教材. http://it-spiral.ist.osaka-u.ac.jp/project/education.html.