平成 24 年度

筑波大学情報学群情報科学類

卒業研究論文

題目

ファイルの関連性を視覚化し、ユーザの操作を記憶するアイコンインタフェース

主専攻　　ソフトウェアサイエンス主専攻

著者　　謝 湘平

指導教員　田中二郎　志築文太郎　高橋伸　三末和男

要　旨

今日のコンピュータの操作は GUI(Graphical User Interface) を介して行うことが一般的である．しかし，GUI においてはアイコンの増加によってデスクトップやディレクトリが煩雑になる，そして，CLI(Command Line Interface) に比べて複雑な操作を行いにくいという問題がある．

本研究では，GUI の利便性を向上させるために，ファイルの操作や内容の関連性を取り入れたアイコンインタフェースを提案し，またそのシステムを実装した．

提案するインタフェースでは，ユーザが以前に行ったファイル操作を記憶し，再現することが可能である．これにより，ユーザは同様の操作を他のファイルに対して再利用することができるため，効率良くファイルを操作できる．また，ファイル名や属性から関連するファイルを検出し，アイコンをまとめることで関連性を視覚的に示す．関連性に基づいたアイコンのまとまりにより，ユーザの目的のアイコンを探す手がかりや，ファイルの内容を予測する手掛かりが増える．

目 次

図 目 次

第1章　はじめに

1.1　GUIについて

今日のコンピュータシステムはGUI（Graphical User Interface）を介して操作することが一般的である．GUIはデスクトップメタファというコンセプトに基づいて設計され，ウィンドウ・アイコン・メニュー及びポインティングデバイスを構成の基本要素とする[1][2].

GUIでは，操作対象がアイコン等のグラフィカルな表現により示され，ポインティングデバイスによる直接操作が可能である．

1.2　ファイルシステムのGUIにおける問題点

1.2.1　見た目における問題点

コンピュータのファイル数の増加に伴って，大量のアイコンがデスクトップ上やディレクトリ内に存在するようになる．一般的なコンピュータユーザは，念入りなファイリングを好まず，ディレクトリ階層を多用してファイルを編成しようとはしない[3]ため，デスクトップやディレクトリはファイルのアイコンによって煩雑になる．それにより，ユーザは目的のアイコンの位置やファイルの内容を把握しにくくなる．

1.2.2　操作における問題点

GUIに対して，コマンド入力に基づく操作のユーザインタフェースはCLI（Command Line Interface）と呼ばれる．

CLIでは，過去使用したコマンドを再利用する，コマンドを組み合わせて一連の操作を一括して処理する，などといった複雑な操作を行うことができる[3]．それに比べ，GUIでは複雑な操作を行いにくい．

たとえば，５つのファイルに対して，それぞれのPDF化を行いたい場合，コマンド入力を用いるとコマンド１行で行うことも可能である．それに対し，アイコンを用いると５つのファイルそれぞれを開いて同じ操作を繰り返す必要がある．

1.3　本論文の構成

第１章では，既存のデスクトップやファイルアイコンの GUI について説明し，問題点を挙げた．

第２章では本研究の目的とアプローチを述べる．第３章では本研究が提案するインタフェースについて説明し，利用シナリオの例を挙げる．第４章では提案するインタフェースを実装したプロトタイプシステムのインタフェースと機能及び操作の仕方について述べ，第５章ではプロトタイプシステムの実装について説明する．第６章では関連研究について述べる．第７章ではまとめと今後の展望について述べる．

第2章　目的とアプローチ

2.1　本研究の目的

本研究ではファイルの操作や内容の関連性をアイコンインタフェースに取り入れることにより既存のファイルシステムの GUI を改善し，ユーザにとっての利便性を向上させることを目的とする．

2.2　本研究のアプローチ

本研究は，ファイル間の関連性を視覚化する．さらにファイル生成時の操作を記憶し，それを使いまわせるような再利用性の高いインタフェースを提案し，そのプロトタイプシステムを設計・実装する．

ファイル間の関連性の視覚化においては、関連性の種類によってアイコンを異なるまとまり方で表現する．そのため，ユーザはアイコンのまとまりによってファイル間に存在する関連性を把握でき，これがファイル内容を思い出す手掛かりにつながる．また，アイコンをファイル間の関連性に基づいて整列させる．そのため，ユーザはアイコンのまとまりを見渡すことによってアイコンの位置を把握できる．これにより，見た目における問題を改善できる．

操作の記憶と再利用を可能にすることで，ユーザは類似した操作を容易に繰り返すことができる．これにより，操作における問題を改善できる．

第3章　提案するインタフェース

3.1　ファイル間の関連性の視覚化

ファイル間には，“ 内容が似ている”，“ ファイル作成時に他のファイルを利用した”といった何かしらの関連性が存在する場合がある．しかし，既存のファイルシステムの GUI では，ファイルのアイコンの一つ一つは独立した状態で表され，ファイル間の関連性は視認しづらい．また，ファイルの名前や属性に基づいたファイルのアイコンの並び替え機能は存在するが，並び替え機能ではアイコンの順番を入れ替えるだけで，アイコンの表示は独立した状態のままである．

本研究では，ファイルシステムの GUI に対してファイル間の関連性を取り入れ，アイコンの表示に関連性を反映させることで，第 1 章で挙げた見た目の問題を改善できると考える．

3.1.1　関連性の定義

ユーザはアイコンを介してファイルの閲覧や編集をする場合が多いため，本研究ではファイル内容と操作に基づく以下の関連性を扱う．

- 類似関係…内容が類似しているファイルが持つ関係
  例：同じ写真でサイズの異なるもの・バージョンの異なる文書ファイル
- 生成関係…ファイルから別のファイルが生成された際に持つ関係
  例：TeX ファイルとそれを用いて作られた PDF ファイル

3.1.2　関連性の視覚化

本研究では関連性を持つファイルのアイコンをまとまりとして配置し，さらにどういう関連性を持つのかを視覚的に示す．これにより，ユーザは従来の GUI では見ることができないファイル間の関連性を見ることができる．さらに関連性に基づいたアイコンのまとまりにより視認性が上がり，見た目における問題を改善する．

3.1.3 まとまりの認識に関する予備調査

アイコンのまとまりを表現するのに，線を引く・囲む・距離を狭める・重ねる等の表現方法が考えられる．本研究では，表現方法の違いによるまとまりの認識の違いに関して予備調査を行い，提案するインタフェースに適切な表現方法を考えた．

目的

線を引く・囲む・距離を狭める・重ねるという 4 つの表現方法のうち，どの表し方が誤認識を招きにくく，まとまりを表現するのに適切であるかを調べるために行った．

被験者

被験者は，12 名の男性で，大学生または大学院生であった．

調査内容

付録 A(pp. 39-40) にある用紙を用いてアンケートを行った．用紙には，線を引く・囲む・距離を狭める・重ねるという 4 種類の効果の内，いずれかの効果が付けられた長方形の集まりを示した図を 14 種類掲載した．ユーザに先入観を与えないために，図の掲載順はランダムとした．

被験者には，その長方形の集まりを見て，何らかのまとまりや関連性を感じたところを自由に囲んでもらった．

調査結果

効果を付けず，等間隔で長方形を配置したものは 12 人全員がまとまり・関連性を感じないと答えた．

線を引いたもの (図 3.1 左)，距離を狭めたもの (図 3.1 中央)，重ねたもの (図 3.1 右) に関しては，全員が同じ囲み方を示した．また，線の形・狭める長方形の数にかかわらず，全員が間に線を引いたもの，間隔の狭いもの，重ねられたものには関連性があると感じた．

囲んだもの (図 3.2 左と図 3.3 左) に関しては，被験者によって囲みに違いがあった．枠で囲まれた長方形を囲む被験者のほかに，色・枠の形が同じだからという理由に基づいて，それらを更に大きい枠で囲む被験者もいた (図 3.2 と図 3.3).

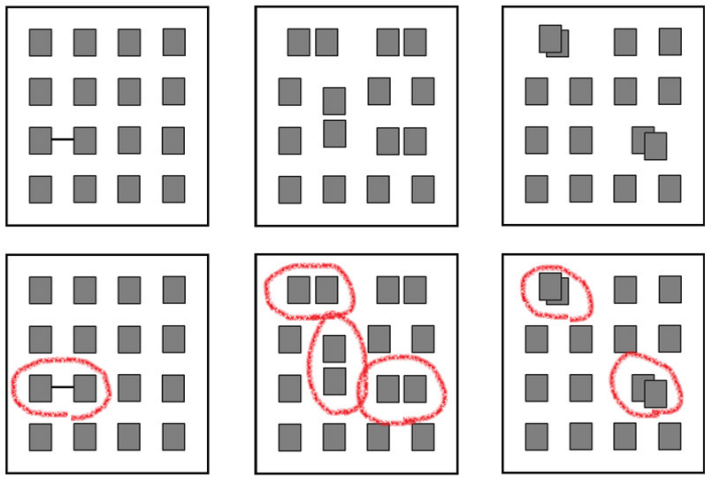


図 3.1: 線 (左)・距離 (中央)・重ねる (右) 表現方法に対するまとまりの囲み方
(上段：元の図，下段：被験者が囲んだまとまり)

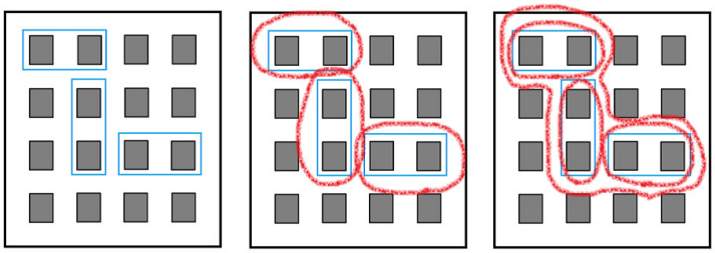


図 3.2: 色付き枠による表現方法に対するまとまりの囲み方
(左：元の図，中央：被験者 7 人の囲み方，右：被験者 5 人の囲み方)

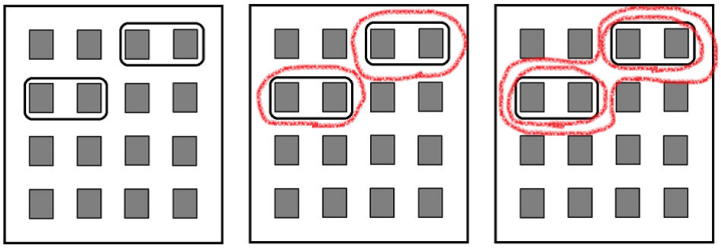


図 3.3: 枠による表現方法に対するまとまりの囲み方
(左：元の図，中央：被験者 10 人の囲み方，右：被験者 2 人の囲み方)

考察

線を引く・距離を狭める・重ねる表現方法に関しては，被験者による囲み方の差が出なかった．また，線は波線や点線などの形の違いによらず，まとまりとして認識された．距離を狭める表現手法も，距離を狭める長方形の個数によらず，まとまりとして認識された．

囲むという表現方法は，同色・同形で囲まれたグループ間にも何らかの関係があるように認識される可能性があり，見る人によって差が出ることがわかった．

以上のことから，誤認識を防ぐには線を引く・距離を狭める・重ねる表現方法を用いた方が良いことが言える．この 3 つのうち，重ねる表現方法は重ねた部分が見えなくなるため，下に位置するアイコンが見づらくなる．また，近接しているもの同士はまとまって認識されやすいことが分かっている[*1]．そのため，本研究には線を引く・距離を狭める表現方法が適切であると考えた．

3.1.4　関連性に基づいたまとまりの表現

関連性に基づいてファイルのアイコンをまとめる．まとまりに異なる見せ方を与えることで，まとまりの階層構造を可能にする．3.1.3 節の予備調査の結果に基づいて，関連性ごとの表し方を決めた．

類似関係

類似関係を持つファイルを内容の類似度の高さによりグループ化する．

予備調査により距離を狭める方法ではユーザによる認識の差が出にくいことがわかった．また，デスクトップ画面では，ユーザは関連性のあるアイコンを，正方形・長方形・円形などの対照的な形になるようにまとめて近くに配置することで，アイコンやファイルを見つけやすくすることが多い [12].

従って本研究では，グループ内のアイコンの間隔を狭くすることにより，まとまりとして表現する方法を取る．また，アイコンの見やすさを考慮し，グループのまとまりをできるだけ正方形に近づける (図 3.4)．図において，長方形はアイコンを表す.

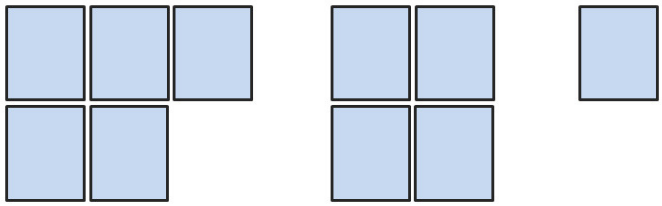


図 3.4: まとまりによる類似関係の表現方法

[*1]プレグナンツの法則における近接の要因

図 3.4 では，左に位置する 5 つのアイコンが 1 つのグループ，中央に位置する 4 つのアイコンが 1 つのグループであることを示している．またグループ間の間隔を広げることで，左の 5 つのアイコンからなるグループ・中央の 4 つのアイコンからなるグループ・右の 1 つのアイコンがそれぞれ独立し，関連性がないことを示す.

生成関係

予備調査の結果から，線で結ばれているアイコンは結んでいる線の形にかかわらず，まとまりとして認識されることがわかった．従って本研究では，矢印をアイコン間に引く表現方法を取る.

生成関係を持つファイルは，矢印を生成元のファイルのアイコンから生成されたあとのファイルのアイコンに向けて伸ばすことにより表現する (図 3.5).

図 3.5 では左のアイコンが示すファイルから右のアイコンが示すファイルが生成されたことを示す.

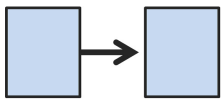


図 3.5: 矢印による生成関係の表現方法

関係の共存

類似関係を持つグループ内に，生成関係を持つものが存在する場合，これを関係の共存と呼ぶ。このとき，生成関係を持つグループと関連するアイコンを近くに配置し，同じまとまりに見えるようにすることにより，関係の共存を表現する (図 3.6).

図 3.6 では，上の 2 者間に生成関係が存在し，さらに 5 つのアイコンが類似関係を持つことを示す.

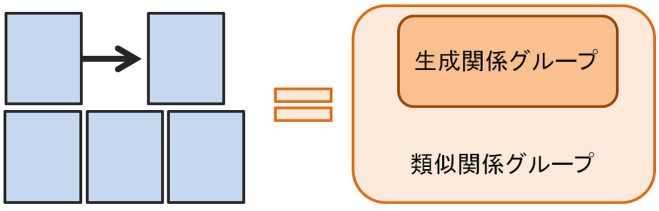


図 3.6: 類似関係の中に生成関係がある場合の表現方法

3.2　関連性に基づく操作のサポート

ユーザは，類似したファイルに対して同じような操作を行うことが多い．その際，対象ファイルに対して毎回同じ操作を繰り返す必要がある．そこで，GUI において操作の再利用を可能にすることで，操作における問題を改善する．

本研究では，システムがユーザの行った操作を記憶し，矢印で記憶した操作を表現する．そのため，ユーザは類似した操作を一から繰り返す必要がなく，矢印を用いることで操作を再利用できる．具体的には以下のように生成関係を持つファイルの更新・生成操作をサポートする．

3.2.1　操作のサポート機能

ファイルの自動更新

生成関係を持つファイル間において，ユーザが生成元のファイルを変更した場合，生成されたあとのファイルにも変更を適用し，自動的に最新版に更新する．つまり提案するインタフェースでは，矢印で示される生成関係は維持される．

記憶した操作の再利用

表示されている生成関係の矢印を他のファイルのアイコンにドラッグ&ドロップすることで，矢印の示している生成操作を，ドロップ先のファイルに対して同様に行うができる (図 3.7).

図 3.7 では，上段に生成関係を持つファイルのアイコンが存在し，アイコン間の矢印を下段にある独立している要素にドラッグ&ドロップすることで，ドロップされた要素に対して同様の操作が行われ，ファイルが自動的に生成されたことを示す.

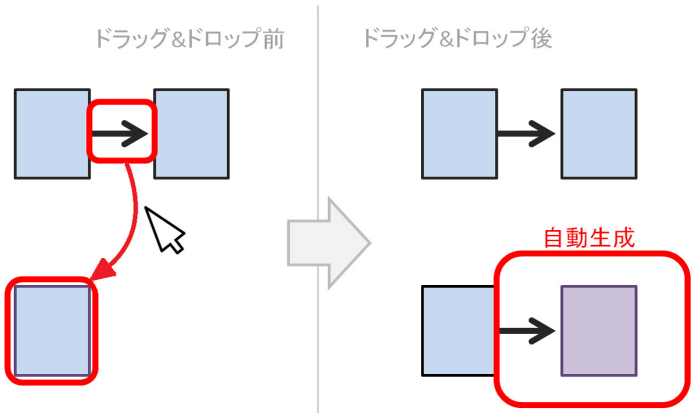


図 3.7: ドラッグ&ドロップによる操作の再利用

3.3　提案するアプローチによる利点

3.3.1　関連性の視覚化による利点

視認性が向上する

関連するファイルのアイコンを近くにまとめ，関連性が視覚化されていることで，ユーザはあるファイルと関連するファイルを素早く把握することができる．

また，アイコンを一つ一つ見るかわりに，まとまりによってアイコンをグループとして見渡すことができる．

関連性によるアイコンの整列ができる

関連性によるアイコンのまとまりにより，従来の名前や属性による並び替えの替わりに，関連性を用いてアイコンを整列することができる．

階層構造が軽減される

関連するファイルのアイコンがまとめられているため，アイコンのまとまりをディレクトリの替わりに用いることで，ディレクトリの多重階層の乱雑さが軽減される．

3.3.2　操作のサポートによる利点

操作を思い出す必要がない

ユーザが一度行ったファイル生成時の操作をシステムが記憶するため，ユーザは操作のやり方や手順をすべて思い出す必要がない．

操作を容易に繰り返せる

システムは記憶したユーザのファイル作成時の操作を再現できる．そのため，ユーザは他のファイルに対して同じ操作を行いたい場合，毎回同じ操作を繰り返す必要がない．操作を意味する矢印をドラッグ&ドロップすることで，ファイル作成を容易に行うことができる．

更新操作を行う必要がない

生成関係にあるファイルは，生成元のファイルの編集履歴に従って更新される．そのため，ユーザは特定の操作を行う必要がない．

3.4 利用シナリオ

ファイル間の関連性が見えることを活かした利用シナリオ

K さんは大容量のコンピュータを使用している．K さんはまめにファイル整理をしない性格であるため，コンピュータのデスクトップやドキュメントのディレクトリはあっという間にファイルのアイコンが溜まり，乱雑になってしまった．

K さんが卒業研究の最終報告書を編集しているとき，ふと以前作成した中間報告書の内容を活用できるのではないかと思い付いた．そこで K さんは本研究のシステムを起動し，編集中の最終報告書ファイルのアイコンに目を向けた．最終報告書ファイルのアイコンの近くには報告書関係のアイコンのまとまりができており，K さんはその中に中間報告書のアイコンを見つけることができ，無事中間報告書の内容を最終報告書にも活かすことができた．

このシナリオのように，本研究のシステムでは関連性を持つファイルをアイコンのまとまりとして表現することで，ユーザはデスクトップやファイルマネージャの端から端までアイコンをチェックすることなく，アイコンのグループとしてまとまりを見渡し，欲しいファイルを見つけることができる．

ファイルへの操作の再利用を活かした利用シナリオ（１）

X さんはコマンドを覚えるのが苦手である．TeX ファイルから PDF ファイルを作成するとき，X さんは作る度に“ platex ”コマンドと“ dvipdfmx ”コマンドをインターネットで検索して使用していた．X さんは本システムの存在を友達伝いに聞きつけ，早速システムを起動した．そこでは X さんが以前作った卒論第 1 稿.tex，卒論第 1 稿.dvi と卒論第 1 稿.pdf の関係が矢印によって示されていた． X さんが表示されている矢印を現在作成中の卒論完成版の TeX ファイルのアイコンにドラッグ＆ドロップすることで卒論完成版の DVI ファイルと PDF ファイルが自動的に生成された．こうして X さんはコマンドを検索することなく PDF ファイルを手に入れられた．

このシナリオのように，ユーザはかつて行ったファイル操作を，矢印のドラッグ&ドロップを用いて他のファイルに対しても行うことができる．

ファイルへの操作の再利用を活かした利用シナリオ（２）

P さんは先生に研究説明の流れをチェックしてもらうために，早急に今月作成した進捗資料の４つ PPT 資料を wiki に載せる必要があった．wiki には PDF ファイルしか載せられないため，P さんは４つの PPT 資料を PDF ファイルに変換しなければいけない．P さんは本システムの“ 操作の再利用 ”機能を思い出し，本システムを起動した． １つ目の PPT 資料を Microsoft PowerPoint 2010 ソフトを用いて PDF ファイルを作成すると，本システムは自動的に１つ目の

PPT 資料と生成された PDF ファイルを矢印で生成関係に結んだ．P さんが表示された矢印を残りの３つの PPT 資料にドラッグ&ドロップすると，残りの３つの PPT 資料についても PDF ファイルが作成された．こうして P さんは４つの PDF ファイルを手に入れ，wiki にあげることができた．

このシナリオのように，ユーザは矢印のドラッグ&ドロップによって，同様の操作を効率よく行うことができる．

第4章　システムのインタフェース設計

4.1　システムの概観

本研究では，提案するインタフェースをファイルマネジャー上に実装した．図 4.1 に今回提案するインタフェースを実装したプロトタイプシステムの概観を示す．

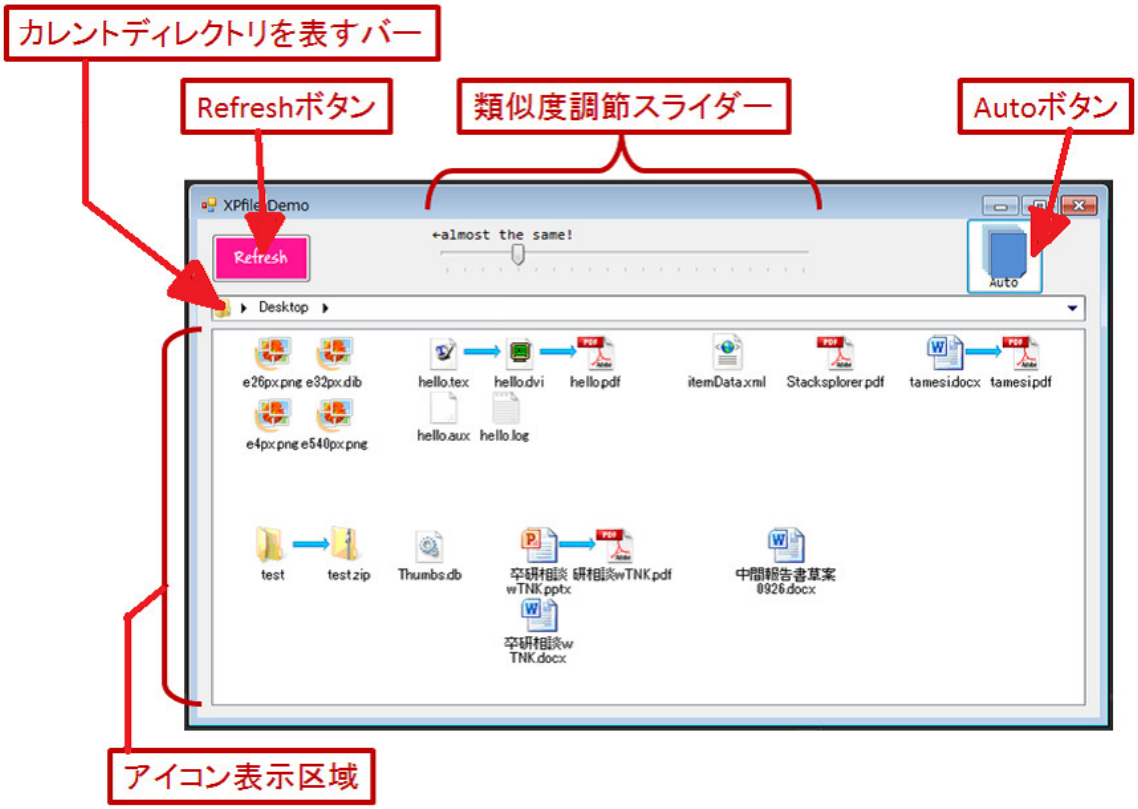


図 4.1: システムの概観

以下に各要素の説明をする．

Refresh ボタン　関連性を表示せずに，等間隔に並んだアイコンを表示するボタン．これを押すことで，ユーザは関連性の表示されない状態のアイコンを見ることができる (図 4.2).

類似度調節スライダー　ファイル間の関連性の計算に用いる類似度を調節するスライダー．アイコンのグループ化を行う際に，スライダーから得られる類似度のしきい値を用いる．左に移動させるとグループ化の条件が厳しくなる．そのためグループに属するアイコン数が少なくなり，まとまりが小さくくる (図 4.3 上)．右に移動させるとグループ化の条件が緩くなる．そのためグループに属するアイコン数が多くなり，まとまりが大きくなる (図 4.3 下)．

Auto ボタン　関連性に基づいてアイコンの自動整列を行うボタン．アイコンの表示を関連性が視覚化されたものにする．また，Auto ボタンを押すことで，ユーザ自身によってカスタマイズされたグループをリセットできる．

カレントディレクトリを表すバー　カレントディレクトリがどこであるかを表示するバーである．

アイコン表示区域　アイコンを表示するエリアである．アイコン数が増えると右側や下にスクロールバーが出る．

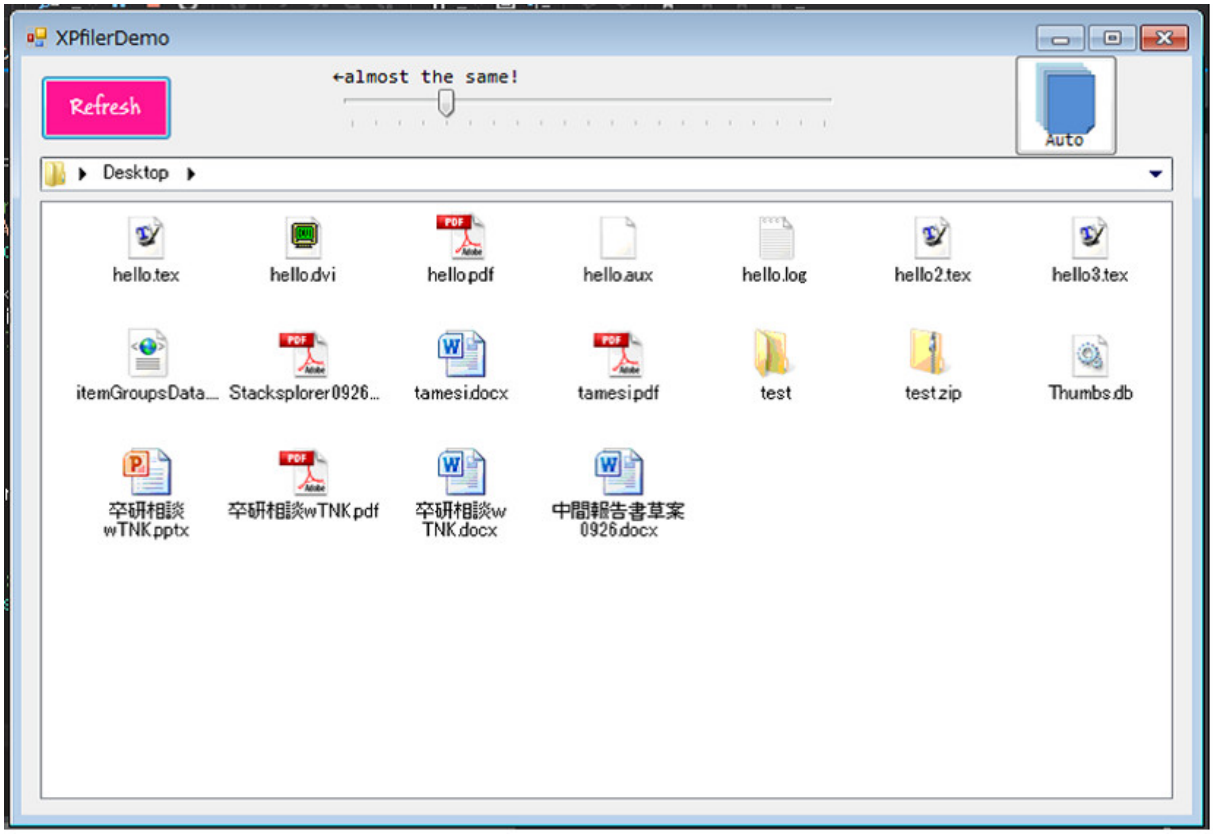


図 4.2: Refresh ボタンが押された状態 (関連性を表示しない状態)

図 4.2 は Refresh ボタンが押された状態を示している．ファイル間の関連性は示されなく，アイコンは名前順に間隔が等しく並べられている．

図 4.3: しきい値の変化によるまとまりの違い (上：しきい値=7，下：しきい値=13)

ファイル間の類似度を用いてアイコンをグループ化する．その際，類似度に課せられたしきい値によってグループ化の度合いを決める．図 4.3 はしきい値の違いによってまとまりの大きさが異なることを示している．しきい値が小さいとき，グループ化の条件は厳しい．そのため図の上側のように関連性が強く，同じ画像の画素値が異なるものがグループ化され，緑色に囲まれた画像のアイコンのまとまりができる．それに対して，しきい値を大きくするとグループ化の条件が緩くなる．そのため画像に少しでも関連性があればアイコンはまとまり，

図の下側のように緑色に囲まれたまとまりが大きくなることが分かる．

4.2　関連性の視覚化

3.1.4 節の表示方法に従って，各関連性の表示を実装した．

4.2.1　類似関係の例

類似関係ではアイコンの間隔を狭くすることで表している．

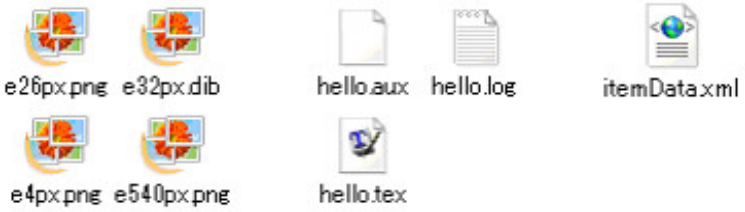


図 4.4: 類似関係の例

図 4.4 では左の 4 つのファイルが類似関係のグループとしてまとまっている．また，中央の 3 つのファイルが類似関係のグループとしてまとまっており，グループ間に関連性がないことを示している．

左の 4 つのファイルは，それぞれ画素値の異なる同じ写真であるため，同じグループに属する．中央の 3 つのファイルは TeX ファイルから PDF ファイルを生成する際に生じるファイルであるため同じグループに属する．右のファイルはどれとも関連がないため独立する．

プロトタイプシステムでは，ユーザにとって見やすいように，類似関係のまとまりをなるべく正方形に近づけている．

4.2.2　生成関係の例

生成関係では関係を持つファイルのアイコン間に矢印を引くことで表している．

図 4.5: 生成関係の例

図 4.5 では，tamesi.docx ファイルから tamesi.pdf ファイルが生成されたことを示している．プロトタイプシステムでは現在のところ，TeX ファイルの PDF 生成，Microsoft Office ファ

イルの PDF 生成，フォルダの ZIP 圧縮フォルダ生成を表示できる．TeX ファイルの PDF 生成では，TeX ファイルから DVI ファイルが生じ，DVI ファイルから PDF ファイルが生成されるため，三者間の生成関係になる．

4.2.3　関係が共存する例

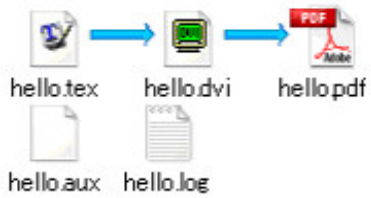


図 4.6: 類似関係内に生成関係を含む例

図 4.6 では類似関係と生存関係が共存する場合の例を示している．類似関係内に生成関係を含むとき，関係の共存が生じる．

全体の類似関係のグループは，hello.tex を hello.pdf に変換した際に生じた 5 つのファイルで構成されている．その中で，hello.tex から hello.dvi が生じ，hello.dvi から hello.pdf が生成されている．そのため，hello.tex と hello.dvi 及び hello.pdf は生成関係を持ち，各ファイルのアイコン間に矢印が引かれている．

4.3　ファイル操作のインタラクション

4.3.1　操作に基づくファイルの自動更新の例

生成関係における生成されたファイルの自動更新の例を以下に示す (図 4.7).

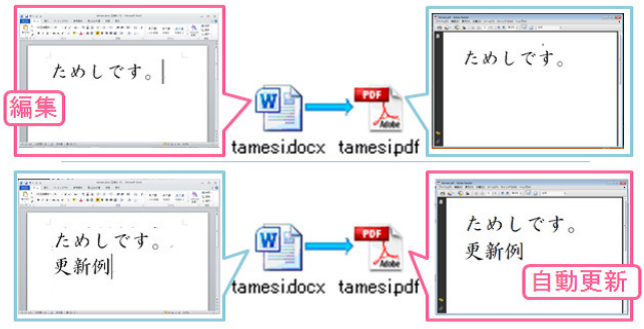


図 4.7: 生成元のファイルの変更に応じた自動更新

図 4.7 の例では，上段が編集前の docx ファイルと PDF ファイルの内容を示し，下段が編集後の各ファイルの内容を示している．

例では，ユーザは temesi.docs ファイルに対して「更新例」という一文を追加する．そのあと，関係している tamesi.pdf ファイルを開くと，編集した内容が PDF ファイルに反映され,「更新例」という一文が PDF ファイルにも追加されていることを確認できる．

4.3.2　記憶した操作の再利用の例

矢印のドラッグ&ドロップによる，ファイル生成操作の再利用の例を図 4.8 に示す.

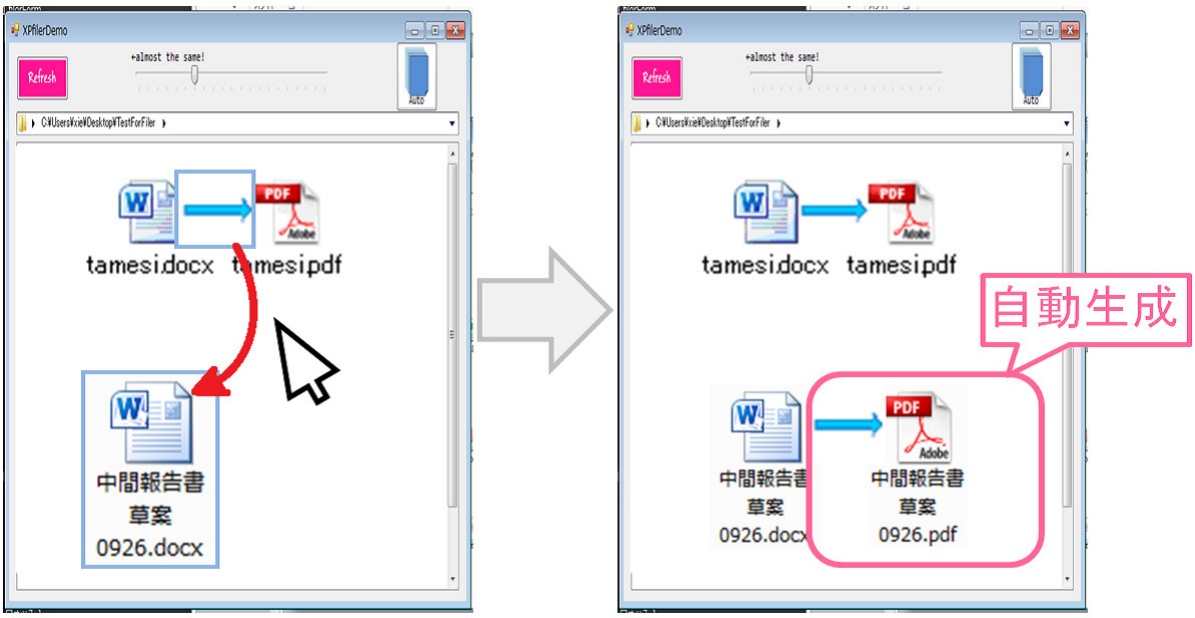


図 4.8: ドラッグ&ドロップによる操作の再利用の例

図 4.8 では，PDF ファイルの自動生成の例を示している．

ユーザが中間報告草案 0926.docx ファイルから PDF ファイルを作りたい場合，表示されている tamesi.docx と tamesi.pdf ファイルの間にある矢印アイコンを，PDF 中間報告草案 0926.docx ファイルにドラッグ&ドロップすると，自動的に中間報告草案 0926.pdf が生成され，新しく生成関係が表示される．

現在，再利用可能な生成操作は，TeX ファイル及び Microsoft Office ファイルの PDF 変換，ZIP 圧縮操作である．

4.4　グループのカスタマイズ

4.4.1　カスタマイズのインタラクション

ファイルのアイコンをドラッグ&ドロップする，または右クリックで出るコンテクストメニューの項目を選択することによって，ユーザは自分好みのアイコンのグループをカスタマイズできる．ユーザが行うことができる操作は以下の通りである．

グループを作る

グループにしたいアイコンを選択し，右クリックで出るコンテクストメニューの項目を選択することにより，グループ化を行うことができる (図 4.9).

グループに追加する

グループに追加したいアイコンを，すでにまとまりになっているグループ内のアイコンにドラッグ&ドロップすることにより，グループに入れることができる (図 4.10).

グループから出す

グループから抜き出したいアイコンを選択し，何もない場所にドラッグ&ドロップする，または右クリックで出るコンテクストメニューの項目を選択することにより，グループからアイテムを出すことができる (図 4.11).

グループ間の移動

移動したいアイコンを選択し，ドラッグ&ドロップの際に，移動先となるグループのアイコン上にドロップすることにより，グループ間の移動ができる．

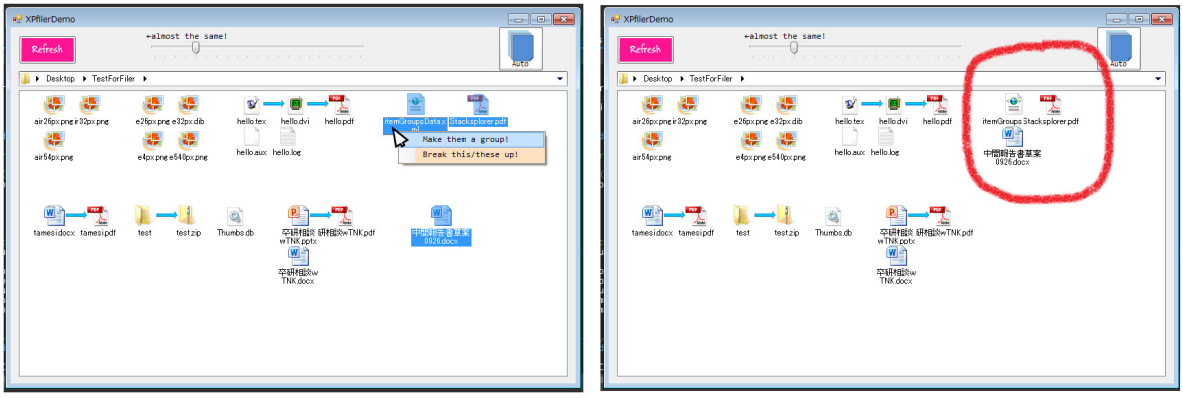


図 4.9: グループを作る例 (左：グループ化したいアイテムの選択，右：グループ化される)

図 4.9 では，3 つのファイルのアイコンを選択し，右クリックで出るコンテクストメニューの項目から「Make them a group」という項目を選択すると (図 4.9 左)，3 つのファイルがグループ化され，アイコンのまとまりが作られる様子を示している (図 4.9 右).

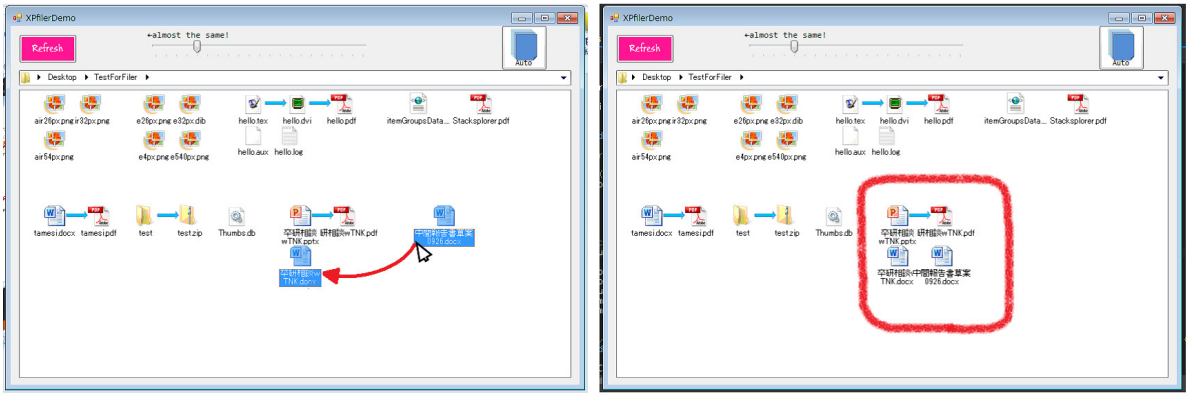


図 4.10: グループに追加する例 (左 : 選択アイテムのドラッグ&ドロップ，右 : グループに追加)

図 4.10 では，ある docx ファイルのアイコンを選択し，すでにグループ化されたアイコンに対してドラッグ&ドロップする操作を示している (図 4.10 左)．これにより，図 4.10 右側のようにドラッグしたアイコンをグループに入れることができる．

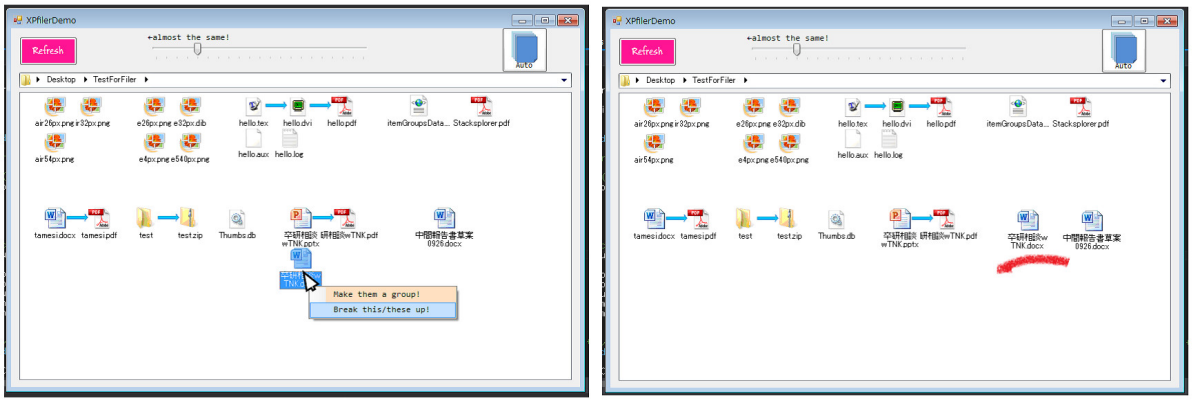


図 4.11: コンテクストメニューの項目選択によりグループから出す例（左 : 選択，右 : グループから出たアイテム）

図 4.11 では，グループに属するアイコンを選択し，右クリックで出るコンテクストメニューから「Break this/these up」という項目を選択することで (図 4.11 左)，アイコンをグループから出すことを示す (図 4.11 右)．

この例では右クリックによるコンテクストメニュー項目を用いているが，その代わりにアイコンをなにもないところにドラッグ&ドロップすることによっても，グループから出すことができる．

4.4.2　カスタマイズされたグループの保存と再現

ユーザによってカスタマイズされたグループは，システムによって記憶される．そのため，ユーザが Auto ボタン (自動整列ボタン) を押してカスタマイズをリセットしない限り，システムを再起動するとカスタマイズされたグループは再現される．

第5章　システムの実装

5.1　開発環境と言語

OS に Windows7 を用いて，開発環境に Visual Studio2012 を使用した．開発言語は C#である．プロトタイプシステムは Windows フォームアプリケーションとして実装している．アイコンの表示には ListView コントロールを用いた．

5.2　自動グループ化処理の流れ

システムを起動すると，自動的にファイルをグループ化し，関連性を視覚化する．その際の処理の流れを記す．図 5.1 は流れを表したフローである．

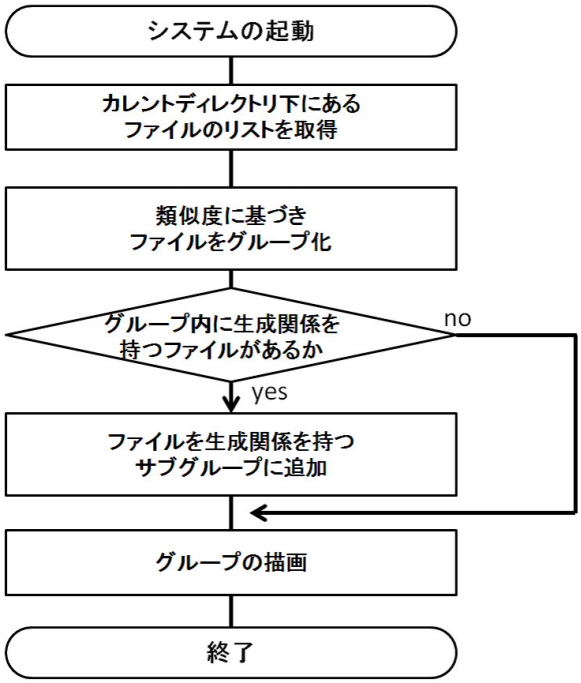


図 5.1: 起動からアイコン表示までのフロー

起動されたとき，システムではまずカレントディレクトリの下にあるサブディレクトリとファイルの一覧を取得する．そのリストの項目を一個ずつ見ていき，ファイル名の文字列の類似度を用いて，ファイルの類似関係によるグループ化を行う (次節で詳細説明).

次に，グループのリストに対して，グループ内のファイルを見比べ，同名のファイルで生成関係に当てはまるものがあるかをチェックする．存在する場合には該当ファイルを生成関係のグループに追加し，存在しない場合には次のグループを見る．

生成関係のグループ作成も完成したら，各関連性の表示方法に従ってグループのアイコンの描画を行う．

5.3 自動的なファイルのグループ化

5.3.1 ファイル間の類似度の計算

ファイル間の類似度の計算に関して，以下の 3 通りの求め方が考えられる．

１：タイトルを分析する

メリット：文書・画像の差が出にくい

デメリット：きちんとしたタイトルが付けられることが前提にある

２：内容を分析する

メリット：精度が高い

デメリット：画像の分析を文書同様にすることが難しい

ファイル数の多さに応じて処理時間が長くなる

３：作成日時やアクセスログを分析する

メリット：キーワードに左右されない

デメリット：コピーや編集操作によって日時が変わる可能性がある

同時にアクセスされたからといって関係があるとは限らない

以上のようなメリット・デメリットを考慮して，今回のプロトタイプではファイルのタイトルに対して，レーベンシュタイン距離を計算して類似度を求める方法を取る．

レーベンシュタイン距離は，一つの文字列を別の文字列に変形する際に，文字の挿入・削除・置換を行う最小回数で与えられる．たとえば，文字列“中間報告書草案”と文字列“最終報告書”のレーベンシュタイン距離は以下のように求められる．[　] の中はレーベンシュタイン距離に足される数字である．

Step1: 中間報告書草案	→	中間報告書草	（1 文字削除）[+1]
Step2: 中間報告書草	→	中間報告書	（1 文字削除）[+1]
Step3: 中間報告書	→	最間報告書	（1 文字置換）[+1]
Step4: 最間報告書	→	最終報告書	（1 文字置換）[+1]

最小の処理回数が 4 であるため，“ 中間報告書草案 ” と “ 最終報告書 ” のレーベンシュタイン距離は 4 となる．

与えられる n 個のファイル名に対して，それぞれ 2 ファイル間のレーベンシュタイン距離を求め，n × n の行列にその値を収める．(n : 任意の数)

次に 3 ファイル間の 3 × 3 行列の例を示す (図 5.2)．例では “ 卒業論文草案 ”・“ 卒論最終版 ”・“ 進捗報告 ” の間のレーベンシュタイン距離を求めている．

	“卒業論文草案”	“卒論最終版”	“進捗報告”
“卒業論文草案”	(0)	7	(10)
“卒論最終版”	(7)	(0)	(9)
“進捗報告”	(10)	(9)	(0)

図 5.2: 3 つのアイテム間のレーベンシュタイン距離を収めた行列

5.3.2 計算量削減と精度向上

計算量削減のために，同一ファイルの比較結果，及び重複する比較結果には無限大を代入する．図 5.2 中の ( ) 付きの文字が無限大を代入するものに該当する．

また，レーベンシュタイン距離を求めるだけでは，間違った関連性が作られる可能性がある．

たとえば，“ 風景 ”・“ 風景_0123 ”・“ 卒論 ” という 3 つのファイル名が存在するとき，レーベンシュタイン距離を求めるだけでは，“ 風景 ” と “ 風景_0123 ” の比較結果 (＝ 5) が，“ 風景 ” と “ 卒論 ” の比較結果 (=4) より小さくなり，“ 風景 ” と “ 卒論 ” の方が関連性があるという結果になってしまう．しかし，実際のファイル名を見るとそうではないことが分かる．そこで，精度を向上させるために，同じ文字を持たないタイトルの比較結果にも無限大を代入した．

5.3.3 類似度に基づいたグループ化

前節の類似度の計算によって得られた n × n の行列をもとに，最長距離法を用いてクラスタリングを行い，n 個のファイルを m 個のグループに分けていく (m : 任意の数)．最長距離法とは，グループ内の要素間の類似度の内，もっとも類似性の低い値をグループ間の類似度と

するクラスタリング手法である．
グループの数と大きさは，クラスタリングの際にしきい値を設けることで調節が可能である．

グループに分ける際の処理の流れを図 5.3 に示す．

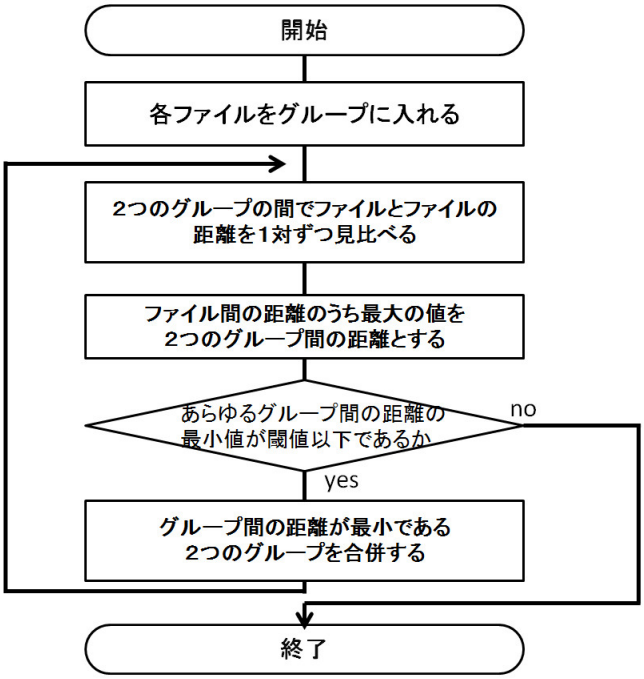


図 5.3: 再長距離法によるグループ化処理のフロー

まず全てのファイルをそれぞれ 1 つのグループとする．次にそれぞれのグループに対して，図 5.3 にあるような流れでグループを合併し，アイテムを追加していく．グループ間の距離の最小値が閾値に達したところでグループ作りを終わりにする．
フォームアプリケーションに設置したスライダーから得られる値をしきい値として，スライダーが動かされる度にグループ化を再度行うようにしている．

5.3.4 生成関係を持つグループの作成

生成関係になり得る拡張子を記した辞書を作成する (5.7.1 節に詳細説明)．類似度によって作られたグループの中で，同じ名前 (拡張子を除く) を持つファイルで，かつ辞書に拡張子が記されたものは生成関係を持つサブグループに追加する．

5.4　カスタマイズによるグループ化

5.4.1　ドラッグ&ドロップを用いる場合

ユーザによるドラッグ&ドロップ操作があった際の流れを説明する．フローチャートを以下の図 5.4 に示す.

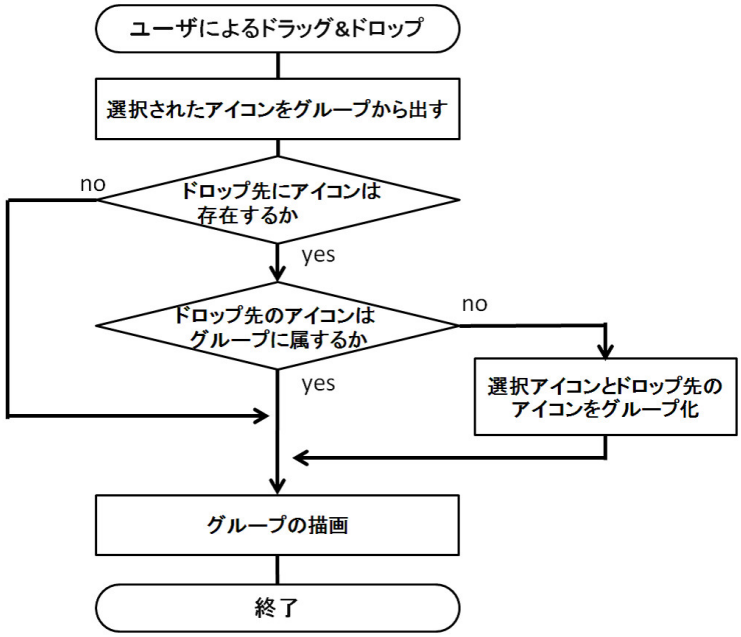


図 5.4: ユーザによるカスタマイズ操作がある際のフロー

選択したアイコンを何もないところにドロップしたとき，選択したアイコンをグループから出す.

ユーザが選択したアイコンを別のアイコンの上にドラッグ&ドロップしたとき，選択したアイコンを現在の所属グループから切り離し，ドロップ先のアイコンの所属するグループに追加する．もし，ドロップ先のアイコンがグループに属していない場合，選択したアイコンとドロップ先のアイコンで新しくグループを作成する.

5.4.2　右クリックによるコンテクストメニューを用いる場合

ユーザが，アイコンを選択し，右クリックによるコンテクストメニューを用いてカスタマイズを行う場合は次のように処理する.

まず選択したアイコンをぞれぞれ現在所属するグループから切り離す．次に新しくグループを作成し，選択したアイコンを入れる.

5.5　グループの保存と再現

フォームアプリケーションのウィンドウが閉じられる際に，グループの状態を XML ファイルに書き出し，情報をシリアライズする．

記録する情報は，以下のようになっている．

- 生成関係を持つグループ：
  - グループの属性
  - アイテムの絶対パス
  - どのアイテムからどのアイテムに矢印が伸びるかの前後関係
  - 生成の際に用いたコマンド
- 生成関係を持たないグループ：
  - グループの属性
  - アイテムの絶対パス

フォームアプリケーションを立ち上げるときに，保存された情報をデシリアライズすることで，前回閉じた時のグループの状態を再現することができる．

情報のシリアライズとデシリアライズにより，ユーザによるカスタマイズを記憶し，カスタマイズされたグループを再現することが可能となる．

5.6　グループの属性と描画

5.6.1　グループの属性

グループには属性をつけ，属性によって描画方法を変える．

属性は以下の 5 種類である．

生成グループ

生成関係を持つアイテムからなるグループである．

各アイテムの絶対パスの他に，どのアイテムからどのアイテムが生成されたかという情報も持つ．

類似グループ

類似関係を持つアイテムからなるグループである．

生成グループを含む類似グループ

類似関係を持つアイテムからなるグループで，グループ内に生成関係を持つサブグループが含まれているものである．

カスタマイズグループ

ユーザのカスタマイズによってできるグループである．

生成グループを含むカスタマイズグループ

ユーザのカスタマイズによってできるグループで，グループ内に生成関係を持つサブグループが含まれているものである．

5.6.2　アイテムの描画

グループは属性によって描画方法が異なる．

“類似グループ”と“カスタマイズグループ”は同じ描画方法を用い，“生成グループを含む類似グループ”と“生成グループを含むカスタマイズグループ”は同じ描画方法を用いる．そのため，生成グループ以外のものは，ユーザには同じようなまとまりに見える．“類似グループ”と“カスタマイズグループ”の違いは，スライダーの値が変更された際，類似度に基づくグループを再計算するときに用いられるかどうかにある．ユーザのカスタマイズによって生じたグループは，カスタマイズを崩さないように，再計算の際に使わない．

実装に ListView コントロールを用いたため，アイコンの表示位置を変えるには，各アイコンに新しい始点となる xy 座標を指定する．

生成グループの描画

生成グループのアイテムの描画では，生成元となるアイテムと生成されたアイテムの間にアイコン一個分の場所を空けて配置し，空けた所に矢印を描く（図 5.5）．

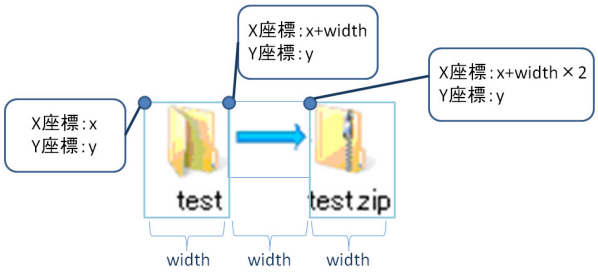


図 5.5: 生成グループのアイコンの座標指定

類似・カスタマイズグループの描画

類似グループ，及びカスタマイズグループのアイテムの描画では，グループ内の最初のアイテムの表示位置を記憶し，その位置からアイコンの幅分横に移動した点に次のアイテムを

置くようにする．

ユーザに見やすいよう，グループをまとまり良く見せるために，まとまりの形を正方形に近づける．

グループのアイテムの個数の平方根をとり，それを切り上げた数値を描画時の改行するまでの一辺の個数とする．たとえば，3 つアイテムがある場合には，sqrt3 を切り上げると 2 になる．その数値に従い，2 つのアイテムを描画したあとに，x 座標を戻し y 座標をアイコンの高さ分ずらした座標を 3 つ目のアイテムの始点にする．

生成グループを含むグループの描画

生成グループを含むグループの描画では，まずサブグループとなる生成グループを描画したあと，生成グループに含まれない他のアイテムを描画する．その際，含まれないアイテムの描画の始点として，生成元となるアイテムの左下のポイントを指定する (図 5.6).

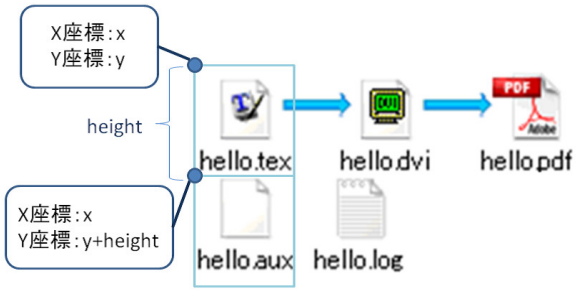


図 5.6: 生成グループを含む場合の座標指定

図 5.6 では，はじめに 3 つのアイテムからなる生成グループを描画する．その後，生成グループに含まれない 2 つのアイテムを描画する．その際，始点として生成グループのアイテムの真下を指定する．

5.7　ファイル操作のサポートの実現

5.7.1　生成操作に関する情報を持つ辞書の作成

生成関係の更新と再利用のために，生成関係を記した辞書を作成する．

生成元アイテムの拡張子・生成されるアイテムの拡張子・生成時に用いるコマンドのリストを集めたものを辞書とする．

5.7.2　操作に基づく更新の実現

生成関係を持つアイテムが変更された際に，生成されたアイテムを更新する処理のフローを図 5.7 に示す.

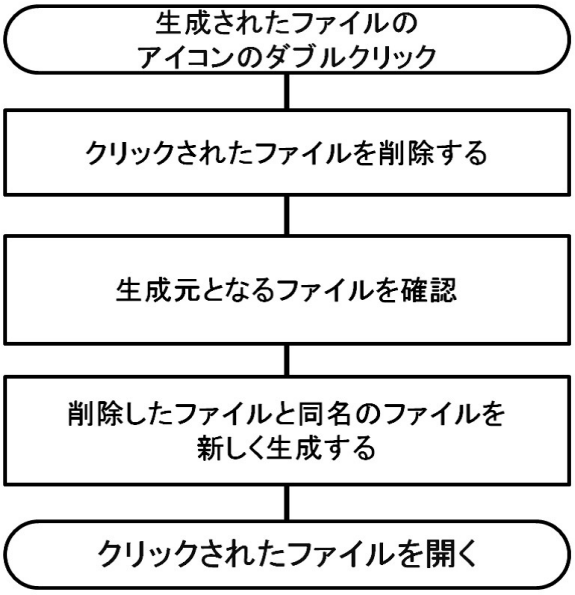


図 5.7: ファイルの更新処理のフロー

ユーザが生成されたアイテムのアイコンをダブルクリックした際，アイテムを開く前に，一度アイテムを消して，新しく作成し直すことで，更新操作を可能にする.

ファイルシステムによって，同じ名前のファイルが作成された際に，片方の名前の後ろに (1) などと番号をつけるものがある．再生成の前に一度アイテムを削除するのは，そういった同名アイテムの重複を防ぐためである.

記憶した外部ファイル (XML ファイル) から読み込んだ，作成時に用いられたコマンドの情報に従って再生成を行う.

5.7.3　操作の再利用の実現

矢印がドラッグ&ドロップされた際に，操作の再利用を実現するフローを図 5.8 に示す.

ユーザが矢印をドラッグ&ドロップした際に，ドラッグした矢印の左側に位置する生成元のアイテムと，ドロップ先のアイテムの拡張子が同じかどうかをチェックする.

同じである場合，辞書を見て，拡張子に対応するコマンドをドロップ先のアイテムに対して実行する．生成されたアイテムとドロップ先のアイテムを生成グループに入れて再描画することで，操作の再利用を可能にする.

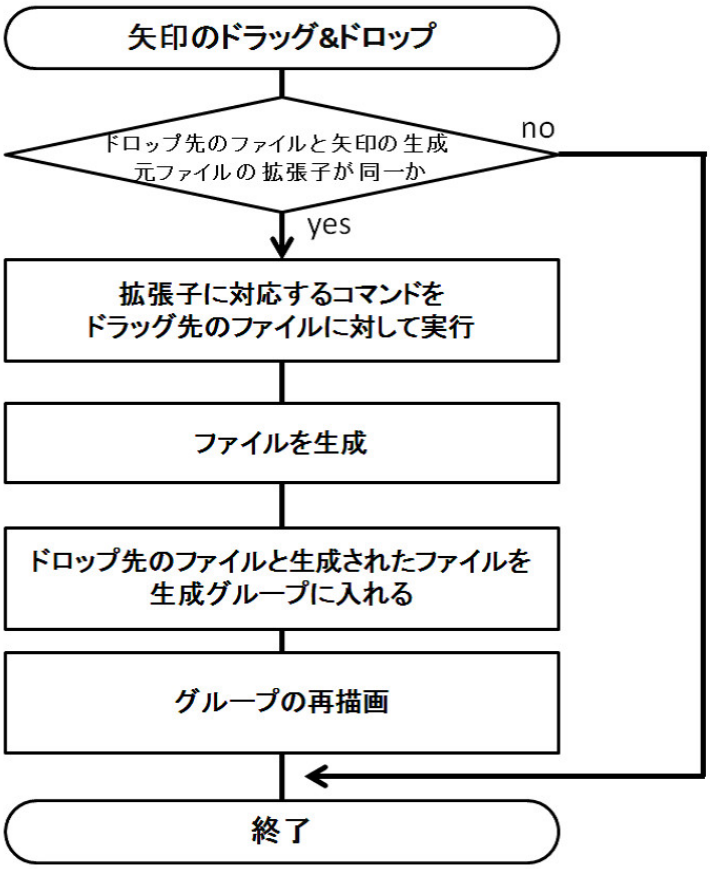


図 5.8: 操作の再利用処理のフロー

第6章　関連研究

後述する関連研究の分類と，本研究の位置づけを示した図を図 6.1 に示す.

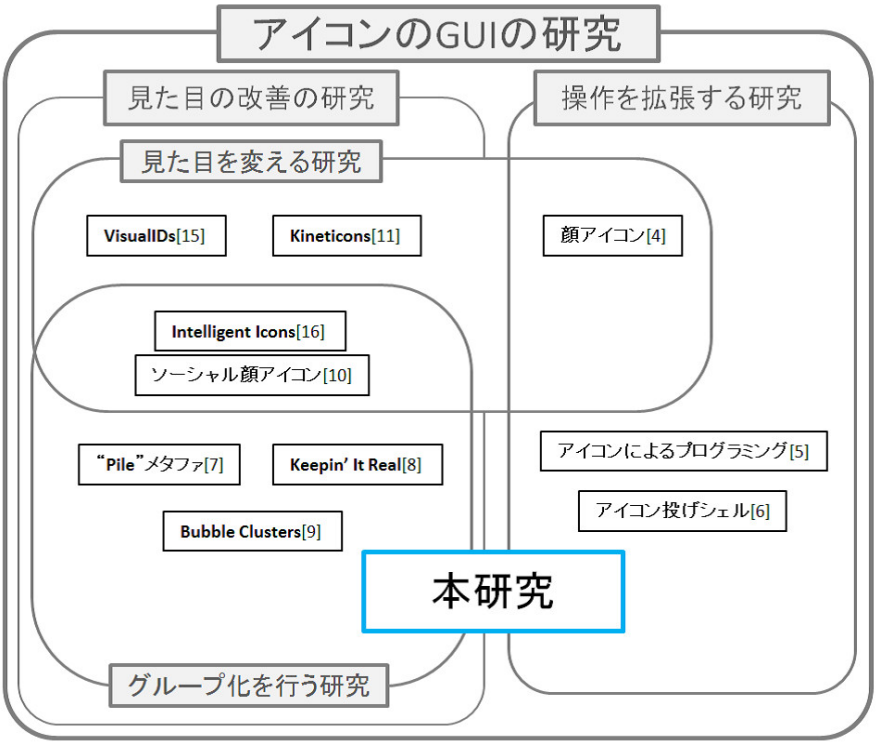


図 6.1: 本研究の位置づけ

6.1　GUI の見た目の改善に関する研究

GUI の見た目の問題を改善するために，アイコンそのものの見た目を変えて，アイコン自体を見やすくする手法と，アイコンをグループ化し，アイコンを見渡しやすくする手法がある.

6.1.1 アイコンの見た目を改善する研究

アイコンの見た目を変えて，ユーザに見やすいようにする研究には，Lewis らの VisualIDs[15] と Keogh らの Intelligent Icons[16] がある．VisualIDs では，ファイル名を分析し，線や円からなる形で表されるアイコンを割り当てている．また，Intelligent Icons では，ファイルの内容を分析し，4 × 4 に区切られた色付きアイコンを割り当てている．

神原らは顔アイコン [4] にソーシャルウェブサイト上のユーザに関連付けたソーシャル顔アイコン [10] を提案した．ソーシャル顔アイコンは，アイコンになっているユーザのソーシャルウェブサイト上での発言をアイコンに付加することにより，アイコンを把握しやすくしている．

Harrison らは Kineticons[11] で，アイコンに動きを取り入れる研究を行った．

アイコン自体を変える手法では，提案されたアイコンがユーザの使い慣れたアイコンに取って代わるため，ユーザが提案されたアイコンに慣れるまでに時間を要すると考える．そのため，本研究ではアイコン自体を変えずに，アイコンをグループ化する手法を取る．

6.1.2 アイコンをグループ化する研究

Mander らはファイルをディレクトリに収めるかわりに，アイコンを積み重ねてまとめる“ Pile ”メタファを提案した [7]．Agarawala らは Keepin’ It Real[8] において，実世界の冊子や書類の重なりに似せて，ファイルを無造作にあるいは整頓して重ねるデスクトップメタファを提案している．

ファイルのアイコンをディレクトリの替わりにアイコンのまとまりによって管理するという点で本研究と Mander らや Agarawala らの研究は関連がある．しかし Mander ら，Agarawala らどちらの研究においてもまとまりの中身となるファイルの収集にはユーザによる操作が必要となる．これに対して，本研究では自動的にシステムによってまとまりを作成する．また，どちらの研究もファイルアイコンのまとまりを重ねあわせによって表現している．これを踏まえ，本研究ではアイコンを重ねずに間隔を狭めることにより，アイコンをユーザに見やすい形にまとめる配置表現を実現した．

Watanabe らの Bubble Clusters[9] では，アイコンの空間的な配置に基づいて自動的にまとまりを作成し，アイコンを取り囲む連結領域によって提示する研究が行われた．

Bubble Clusters は自動的にアイコンをグループ化し，まとまりを視覚的に提示する点においては本研究と関連する．しかし，Bubble Clusters ではファイルのアイコンの画面上の配置の近さに基づいてグループ分類を行ったのに対して，本研究ではファイルの類似度に基づいてグループ分類を行った．これにより，ユーザは従来の GUI では視認できないファイル間の関連性を見ることが可能である．

Watanabe らの Bubble Clusters を踏まえた上で，大河原らはアイコン整理システム [13] を研究し，アイコンのまとまりを作成する際に，ユーザの使い方に応じた自動グループ化方法を提案した．

システムによって自動的にアイコンのグループ化がされる点で本研究と関連する．しかし，

アイコン整理システムはユーザによるファイルアイコンの移動情報に基づいて分類されるのに対し，本研究ではユーザのつけたファイルタイトルの類似度で分類する点で異なり，本研究ではさらに分類されたグループに応じた表示方法も提案している．

6.2　アイコンの操作を拡張する研究

高林らの顔アイコン [4] では，メールのファイル転送を，ファイルを送信相手の顔写真アイコンにドラッグ＆ドロップする操作により可能にした．

一方，本研究では記憶したファイル操作を，矢印をドラッグ＆ドロップするという操作により実現可能にし，さらに GUI におけるファイル操作の再利用性を高めた．

岩田らはアイコンの選択・移動・配置によってプログラミングを行う研究を行った [5]．同様にアイコンを用いてプログラミングを行う考えは平川らの本でも述べられている [14]．

岩田らの研究では，アイコンを用いて画面で対話的にプログラムの作成を行うことができ，記述されたプログラムのデータの流れはアイコンによって視覚的に示されている．

アイコンの組み合わせを用いてコンピュータの操作を行う点で本研究と関連がある．しかし，岩田らの研究ではプログラミングをするためにアイコンを用いているのに対し，本研究では GUI により多くの操作を可能とするためにアイコンを用いている点で異なる．また，岩田らの研究ではプログラムの流れをアイコンによって視覚化するのに対して，本研究ではファイル間の関連性を視覚化する点で異なる．

久野らはアイコン投げ [17] という考えを発展させたアイコン投げシェル [6] において，ファイル操作やプログラムの起動を行う GUI に対し，ドラッグ＆ドロップを高速化した「アイコン投げ」操作を使うことで，GUI によるシェルコマンド操作を可能にしている．

アイコンのドラッグ＆ドロップ操作によってファイル操作をし，新たなアイコンを生成する点では本研究と関連がある．しかし，アイコン投げシェルでは CLI のコマンドをアイコンの形で GUI で活用できるようにしたことに対し，本研究ではファイル間の関連性に基づく操作をアイコンにして，操作に活用できるようにした点で異なる．

第7章　まとめと今後の展望

本研究では，ファイル間の関連性を視覚化し，関連性に基づくファイルの更新と生成操作をサポートするアイコンのインタフェースを提案し，そのプロトタイプシステムを設計・実装した．

これによりユーザはファイル間の関連性を把握し，関連性に基づくファイルのアイコンのまとまりを用いて煩雑なデスクトップやディレクトリ内のアイコンを整列できる．またシステムが関連性とそれに基づくファイル操作を記憶することにより，ユーザは操作を再利用することが可能である．

プロトタイプシステムではファイルの関連性の判断に，ファイル名の文字列の類似度を用いたが，ファイルの内容も利用できるようにしたい．また，関連性に基づいたより多くの操作を行えるようにしたい．そして，定量的な評価を行い，結果に応じた改善を行いたい．

謝辞

本研究を進めるにあたり，指導教員である田中二郎先生をはじめ，志築文太郎先生，高橋伸先生，三末和男先生には，ゼミなどを通して丁寧なご指導や貴重なご意見をいただきました．深く御礼申し上げます.

田中二郎先生にはテーマの選び方，目的の決め方から研究の進め方，論文の書き方に至るまで，始終丁寧かつ熱心なご指導をいただきました．心よりお礼申し上げます.

参考文献

[1] 大谷和利. GUI の起源と進化の概略 (情報とデザイン). 情報の科学と技術, Vol. 49, No. 12, pp. 632-638, 1999.

[2] 久保田晃弘, 宮崎光弘, 永原康史, 松田純一, 大谷和利, 杉山久仁彦. GUI を再発明する–ポスト・ウィンドウ・インターフェイス序論. 多摩美術大学研究紀要, Vol. 23, pp. 163-170, 2008.

[3] 鎌田敏之, 有ヶ谷亮介, 大西研治. デスクトップにおける適応インタフェースの実装と評価. 愛知教育大学研究報告, Vol. 52, pp. 51-57, 2003.

[4] 高林哲, 塚田浩二, 増井俊之. 顔アイコン：手軽なファイル転送システム. インタラクション, 情報処理学会, pp. 33-34, 2003.

[5] 岩田誠司, 吉本岩生, 平川正人, 田中稔, 市川忠男. アイコンプログラムの作成・実行環境の開発. 情報処理学会全国大会, pp. 683-684, 1986.

[6] 久野靖, 角田博保, 大木敦雄, 粕川正充. アイコン投げシェル. 情報処理学会 第 38 回プログラミングシンポジウム報告集, pp. 65-74, 1997.

[7] R. Mander, G. Salomon, and Y. Wong. A 'Pile' Metaphor for Supporting Casual Organization of Information. Proceedings of the SIGCHI conference on Human factors in computing systems(CHI), pp. 627-634, 1992.

[8] A. Agarawala, R. Balakrishnan. Keepin' It Real: Pushing the Desktop Metaphor with Physics, Piles and the Pen. Proceedings of the SIGCHI conference on Human factors in computing systems(CHI), pp. 1283-1292, 2006.

[9] N. Watanabe, T. Igarashi. Bubble Clusters: An Interface for Manipulating Spatial Aggregation of Graphical Objects. InProceedings of the 20th annual ACM symposium on User interface software and technology(UIST), pp. 173-182, 2007.

[10] 神原啓介, 塚田浩二. ソーシャル顔アイコン. コンピュータソフトウェア, pp. 172-182, 2011.

[11] C. Harrison, G. Hsieh, K. D.D. Willis, J. Forlizzi, and S. E. Hudson. Kineticons: Using Iconographic Motion in Graphical User Interface Design. InProceedings of the 20th annual ACM symposium on User interface software and technology(UIST), pp. 1999-2008, 2011.

[12] P. Ravasio, S. G. Schuar, and H. Krueger. In pursuit of desktop evolution: User problems and practices with modern desktop systems. ACM Trans. Comput.-Hum. Intract., pp. 156-180, 2004.

[13] 大河原昭, 坂本大介, 五十嵐健夫. 学習機能をもったアイコン整理システム. 第 19 回インタラクティブシステムとソフトウェアに関するワークショップ論文集 (WISS), pp. 47-52, 2011.

[14] 市川忠男, 平川正人. かわりゆくプログラミング. 共立出版, pp. 65-74, 1994.

[15] J. Lewis, R. Rosenholtz, N. Fong, and U. Neumann. VisualIDs: Automatic distinctive icons for desktop interfaces. In Proceedings SIGGRAPH. ACM Press, pp. 416-423, 2004.

[16] E. Keogh, L. Wei, X. Xi, S. Lonardi, J. Shieh, and S. Sirowy. Intelligent Icons: Integrating Lite-Weight Data Mining and Visualization into GUI Operating Systems. In Proc. of the 6th ICDM, IEEE Computer Society, pp. 912-916, 2006.

[17] 久野靖, 角田博保, 大木敦雄, 粕川正充. 「アイコン投げ」ユーザインタフェース. コンピュータソフトウェア, Vol. 13, No. 3, pp. 38-48, 1996.

付録A　予備調査に用いたアンケート用紙

氏名：　　男・女　　所属：

なるべく考えずに直感的に答えてください.

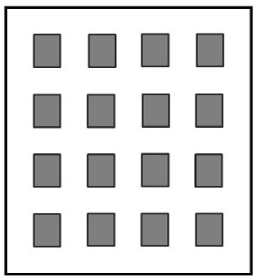
以下に枠に囲まれた長方形の集りが 14 個あります.

枠内にある長方形の中で，まとまりを感じるもの・なんらかの関連性を感じるものはありますか？

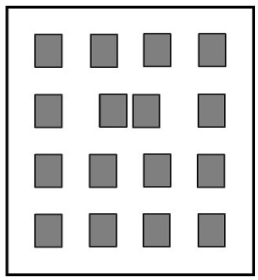
→ある場合には，感じたまとまりを囲んでください．まとまりは複数でも構いません.

→ない場合には枠の番号に×印をつけてください.

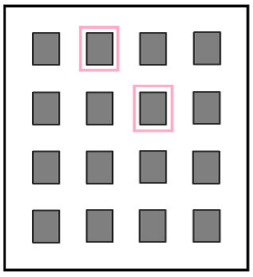
1：



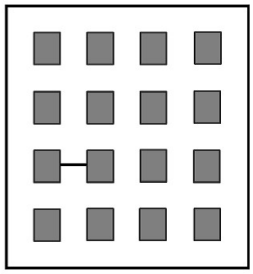
2：



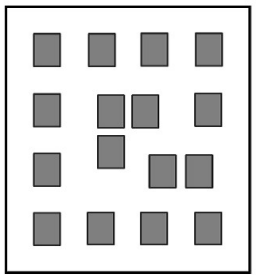
3：



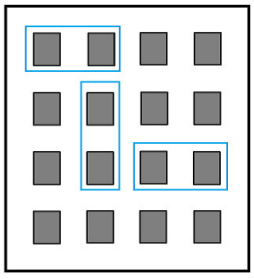
4：



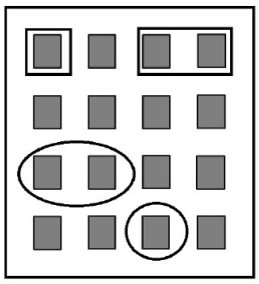
5：



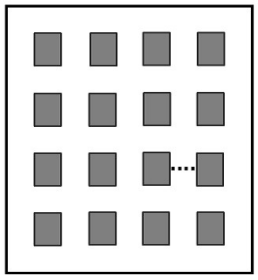
6：



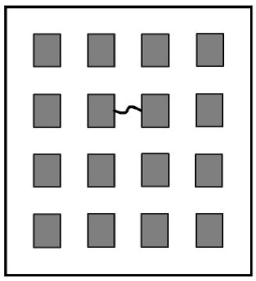
7 :



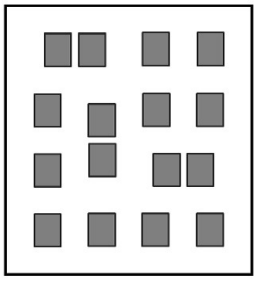
8 :



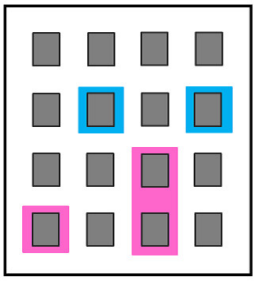
9 :



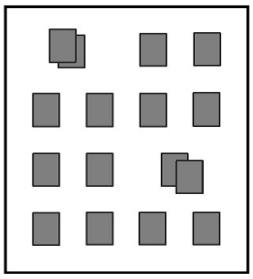
10 :



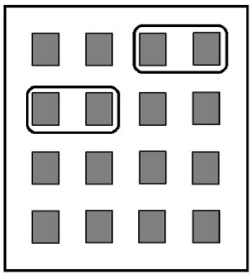
11 :



12 :



13 :



14 :