ソフトウェア利用知識の共有における機能実行履歴の提示

森崎 修司，門田 暁人，松本 健一，井上 克郎，鳥居 宏次

1 はじめに

ワープロ，表計算，作図ソフトなどの一般向けパッケージソフトウェアの多くは，今日，非常に多機能となっている．例えば，Microsoft Word2000 では，メニュー（コンテキストメニューを含む）から実行できる機能だけでも約 500 用意されている．同様に，Adobe Photoshop 5.0 は約 350 のメニュー項目を持つ．これらのソフトウェアは様々な使用状況を想定して開発されているため，多くの状況において必要不可欠となる基本的な機能から，特定の状況においてユーザの作業効率を高める機能まで，様々な粒度の機能が数多く存在し，全体としての機能数が膨れ上がっている．

しかし，機能数の増加は，各機能を学習するためのコスト増大を招くため，ユーザの作業効率の向上に必ずしも結びついていない．提供されている膨大な数の機能を全て学習することは多くのユーザにとって困難である．仮に全ての機能を学習できたとしても，特定の種類の作業のみを行うユーザにとっては不要な機能を数多く学習してしまうことになり，効率的であるとは言えない．

ユーザが行う作業に役立つ機能のみを選択して学習することは効率的であると言えるが，利用できる状況の限られた粒度の大きい機能を学習するコストは高い．一般に，粒度の大きい機能は，粒度の小さな複数の機能で代用できる．そのため，ユーザは，必要最小限の粒度の小さな機能を身に付けた時点で学習を停止する傾向にある [3]．例えば，ワープロソフトを使用した文書作成において，目次の作成，図表番号の指定，指定範囲内のフォント種類の一括置換などの作業は，粒度の大きい機能を用いれば作業効率を著しく高

めることができる場合があるが，ユーザがそれらの機能の存在に気づき，自ら学習するとは限らない．オンラインヘルプによる機能の検索が可能な場合でも，適切なキーワードを選択し，存在すら想像できない機能を見つけ出すことは，多くのユーザにとって容易でない．また，ユーザが存在に気づいている機能であっても，その使用方法が分かりにくい場合には，使用方法をわざわざ学習するよりは，既知の機能の組み合わせで作業を進めることが多いことも観察されている [14]．現状では，粒度の大きい機能は，ユーザがその存在に気づきにくく，しかも，ユーザがその存在に気づいていても利用されないままになる傾向があると言える．

著者らは，ソフトウェアが提供する機能の学習（利用知識の獲得）のコストを小さくすることを目的として，作業内容の類似するユーザ間で，利用知識を共有する方式を提案している [12]．ユーザ間で存在が既知である機能が異なれば，共有によりその数が大きく増えることが期待され，かつ，各ユーザが類似の作業を行っているならば，無駄な知識の獲得も避けられると期待される．文献 [11] では，機能の利用知識がユーザの機能実行履歴にあらわれると考え，ユーザの機能実行履歴を収集し，ユーザ間で相互参照可能とする方式を提案した．評価実験の結果，ユーザ間で利用知識を持つ機能が共有可能であることが確認された．また，実験の結果，機能実行履歴の提示方法がユーザの利用知識獲得に大きな影響を与える可能性があることが分かったが，効果的な提示方法は明らかとなっていない．これまでに提案した機能実行履歴の提示方式は，各ユーザの機能実行履歴に含まれるすべての部分系列（機能実行パターン）とその出現頻度を列挙するものであった．しかし，提示された機能実行履歴から有用な情報を探し出すことは，ユーザにとって必ずしも容易でなかった．特に，存在に気づいていない機能の発見，特定の機能を実行するための前提となる機能の発見，組合せにより効果を発揮する機能の発見など，目的に応じて利用知識を獲得することは困難であった．

本稿では，ソフトウェアが提供する機能の利用知識を共有する際の機能実行履歴の提示方法を提案する．特に，ユーザがその存在に気づいていない可能性のある機能，及び，存在に気づいているが前提となる機能や組合せて利用できる機能の発見を支援する提示方式を提案する．

以降，2 章では機能実行履歴の共有について述べ，3 章で提案する機能実行履歴の提示方式について述べたあと，4 章で試作システムを簡単に紹介す

る．5 章では評価実験，6 章では実験結果について考察する．7 章で関連研究を紹介し，8 章でまとめる．

2　機能実行履歴の共有

2.1　概要

文献 [11] では「利用知識の共有」は同一のソフトウェアを使用するユーザ間で機能の存在や使用方法を相互参照可能にすると定義している．特に，粒度の大きい機能の存在や使用方法を相互参照することにより，作業効率を高めることができる．そのために，ソフトウェアの利用知識をソフトウェアの機能実行履歴として有形化し，似通った作業を実施する複数ユーザの履歴を収集する．収集した履歴に含まれる機能，及び，機能実行パターンに各ユーザの出現頻度などの情報を付加し，ユーザに提示することにより，機能の存在，機能の組合わせ，あらかじめ実行しておくべき機能（前提機能）などの知識を共有することができる．

2.2　機能実行履歴の収集

機能実行履歴の収集，及び，共有の方式の概略を図 1に示す．利用知識を共有しようとするユーザは「機能実行履歴サーバ」とネットワークで結ばれた計算機上で普段どうりソフトウェアを利用する．すると，ユーザの計算機上で稼動する「機能実行履歴収集部」が機能実行履歴を収集し，ネットワークを介して「機能実行履歴サーバ」へと送出する．「機能実行履歴収集サーバ」上で稼働する「実行機能受信部」は，機能実行履歴をユーザ毎に分類した上で，機能実行履歴を蓄積する．「機能実行履歴サーバ」で稼働する「実行機能パターン送信部」はユーザからの要求に応えて，「機能実行履歴解析部」で抽出した機能実行のパターン，パターンの各ユーザにおける実行頻度を集計し，ネットワーク経由でユーザの計算機に送出する．ユーザの計算機上で稼動する「機能実行履歴提示部」では，抽出された実行機能パターンをユーザに提示し，ユーザ間での利用知識の交換を支援する．

機能実行履歴は機能実行履歴収集部がユーザの操作を監視することにより収集する．ユーザが選択したメニュー項目や GUI ボタンのキャプションなど他ユーザが操作を再現可能な文字列に日時を付加して機能実行履歴として記

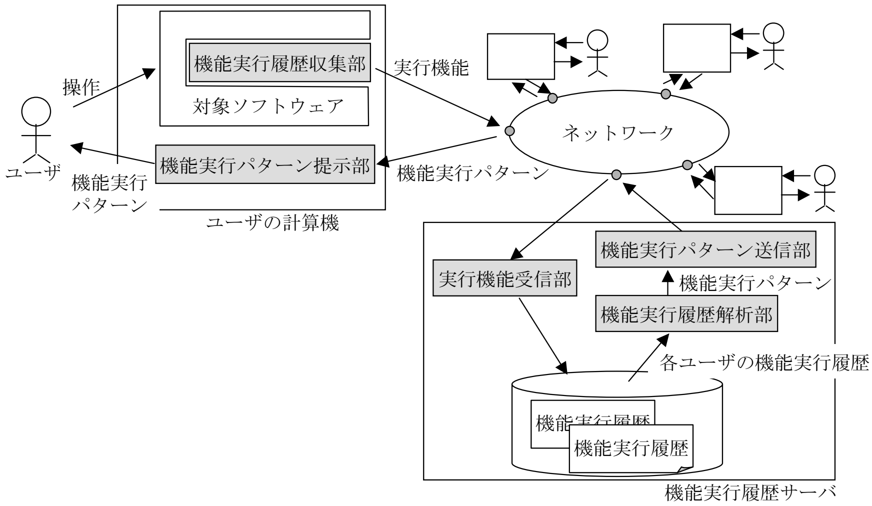

図 1: 機能実行履歴の共有

2000/02/03 18:51:01 ファイル**->**印刷プレビュー**(&V)**
2000/02/03 18:51:14 ファイル**->**印刷 **(&P)**
2000/02/03 18:51:16 ファイル**->**上書き保存**(&S)**
2000/02/03 18:51:23 ファイル**->**終了**(&X)**

図 2: 機能実行履歴の例

録する．機能実行履歴の例を図 2に示す．1 行が 1 つの機能実行に対応し，機能実行毎に実行日時，メニュー項目名，あるいは，ショートカットボタン名が記録されている．メニュー項目名については，トップメニューからの選択経路，ショートカットボタンについてはそのボタンが属するボタンバー（ツールバー）名が付加される．

2.3 機能実行履歴に関する定義

機能実行履歴分析部は，機能実行履歴収集サーバに蓄積された各ユーザの機能実行履歴に含まれる機能実行パターンを抽出するとともに，そのパターンのユーザ毎の出現頻度を調べる．具体的には，全てのユーザの機能実行履歴の集合 $\mathcal{O}$ から出現頻度 1 以上，パターンを構成する機能が 1 以上の機能実

行パターンの集合 $\mathcal{P}$ を求め，各機能実行パターンのユーザ毎の出現頻度 $\mathcal{F}$ を調べる．

$\mathcal{P}$, 及び，$\mathcal{F}$ は，機能実行履歴 $\mathcal{O}$ に含まれる 1 機能 $o_{i,h}$ ，機能実行パターンを構成する 1 機能 $p_{j,k,u}$ ，機能実行パターンの出現頻度 $f_{j,k,u}$ とし，次のとおり定義する．

全てのユーザの機能実行履歴の集合 $\mathcal{O}$:

$\mathcal{O} = \{O_1, \cdots, O_i, \cdots, O_t\}$ (t : 対象となるユーザ数)．

ユーザ i の機能実行履歴 O_i:

$O_i = o_{i,1}, \cdots, o_{i,n_i}$

(n_i: ユーザ i の総機能実行数)．

長さ j の機能実行パターンの集合 P_j:

$P_j = \{P_{j,1}, \cdots, P_{j,k}, \cdots, P_{j,l_j}\}$

(l_j: $\mathcal{O} \in P_j$ である P_j の数)．

長さ j の機能実行パターンの集合 P_j のある要素 $P_{j,k}$:

$P_{j,k} = p_{j,k,1}, \cdots, p_{j,k,j}$

(ただし，$1 \leq s \leq n_i - j + 1$ を満たす s に対して $p_{j,k,1} = o_{i,s}, p_{j,k,2} = o_{i,s+1}, \cdots, p_{j,k,j} = o_{i,s+j-1}$)．

あるユーザ u の機能実行履歴 O_u における $P_{j,k}$ の出現頻度を $f_{j,k,u}$ ($1 \leq u \leq t$) とし，

$F_{j,k} = f_{j,k,1}, \cdots, f_{j,k,u}, \cdots, f_{j,k,t}$

長さ j の機能実行パターンの出現頻度の集合 F_j:

$F_j = \{F_{j,1}, \cdots, F_{j,k}, \cdots, F_{j,l}\}$ ．

機能実行パターンの最大長を m として，機能実行パターンの集合 $\mathcal{P}$，機能実行パターンの出現頻度の集合 $\mathcal{F}$:

$\mathcal{P} = \{P_1, \cdots, P_j, \cdots, P_m\}$

$\mathcal{F} = \{F_1, \cdots, F_j, \cdots, F_m\}$

表 1は抽出するパターンの最大長 m を 2 とし，二人のユーザの機能実行履歴をそれぞれ $O_a = \{$ 機能 $A,$ 機能 $B,$ 機能 $D,$ 機能 $C\}$，$O_b = \{$ 機能 $A,$ 機能 $D,$ 機能 $C,$ 機能 $A\}$ とした場合の例である．

表 1: 機能実行履歴と機能実行パターンの例

O_a	$\{A, B, D, C\}$
O_b	$\{A, D, C, A\}$
P_1	$P_{1,1} = \{A\}, P_{1,2} = \{B\}, P_{1,3} = \{D\}, P_{1,4} = \{C\}$
P_2	$P_{2,1} = \{A, B\}, P_{2,2} = \{A, D\}, P_{2,3} = \{B, D\}, P_{2,4} = \{C, A\}, P_{2,5} = \{D, C\}$
F_1	$F_{1,1} = \{1, 2\}, F_{1,2} = \{1, 0\}, F_{1,3} = \{1, 1\}, F_{1,4} = \{1, 1\}$
F_2	$F_{2,1} = \{1, 0\}, F_{2,2} = \{0, 1\}, F_{2,3} = \{1, 0\}, F_{2,4} = \{0, 1\}, F_{2,5} = \{1, 1\}$

3 提案方式

提案方式では，単一機能の一覧，機能実行パターンの一覧の 2 種類の提示方法により，それぞれ，ユーザが存在に気づいていない機能の発見，及び，ある機能と組合せて実行する機能やある機能の前提となる機能，の発見を支援する．存在が既知でない機能については，単一機能だけを提示し，前提や組合せ機能に関してはユーザが存在に気づいている機能を中心として，その前後に実行された機能を提示することにより，そのコンテキストを提示する．

3.1 単一機能の一覧

単一機能の一覧は，ユーザが存在に気づいていない機能を発見することを支援することを目的とし，1 機能（長さ 1 の機能実行パターン P_1）を表形式でユーザに提示する．表の各行は 1 機能に対応し，各列は以下の項目を含み，機能名以外の列で昇順，降順に並べ替えることができる．

- 機能名（P_1）
  機能を実行するための GUI メニューやショートカットボタンのキャプション．これを見たユーザが操作を再現できるようメニューの場合にはルートメニューからの選択経路，ショートカットボタンの場合には，ショートカットボタンが属するボタンバー（ツールバー）の名称を含める．ユーザは，ルートメニュー，ボタンバーの名称の表示，非表示を切替えることができる．

- 他ユーザの実行割合（ $count(f_i \geq 1)/$ t ）

  機能を 1 度以上実行したことがある自分以外のユーザの割合

- 全ユーザの実行頻度（ $\Sigma F_{j,k}$ ）

  全ユーザの機能実行履歴における機能の出現頻度

- 各ユーザの実行頻度（ $f_{j,k,u}$ ）

  各ユーザの履歴における出現頻度とその割合

ユーザは，機能名から機能自体は未知であっても，自分が使っている機能よりも粒度が大きい可能性のある機能であることが推測できれば，その機能により作業効率を高めることができる．例えば，自分が頻繁に「コピー」「ペースト」を繰返している場合に，「連続データのペースト」という機能名の提示だけで，その機能がより作業効率を高める可能性のある機能であることが判断できる．また，機能名を検索キーワードとしてオンラインヘルプを参照したり，実際に実行してみたりして，その機能がどのようなものであるか知ることができる．

他ユーザの実行割合から，大半の他のユーザが実行したことがあり，自分が実行したことがない機能を知ることができる．大半のユーザが実行したことがあり，自分が実行したことがない機能の場合には，その機能が自分が知っている機能よりも作業効率を高める機能である場合がある．特に，このような機能と自分が実行したことがあり，他ユーザが実行したことがない機能とが同一の目的を果たすことができる場合がある．

全ユーザの実行頻度から，その機能がどの程度実行されているかを知ることができる．機能により実行される頻度は異なることが考えられるが，その機能が頻繁に実行されるものか，そうではないものかにより，機能の内容を推測できる場合がある．

各ユーザの実行頻度から，ユーザによる実行の偏りを知ることができる．他ユーザの実行割合は 1 度以上実行されたことしかわからないが，各ユーザの実行頻度を見ると，実際には同一ユーザだけが頻繁に実行していることがわかる場合があったり，各ユーザが均等に実行している機能であることがわかったりする．特に，自分が頻繁に実行していて，不便だと感じている機能をほとんど実行していないユーザが実行している機能を見ることにより，より作業効率を高める可能性のある機能を発見することができる．

ソフトウェアが提供する機能が膨大であれば，単一機能の一覧も膨大になるおそれがあるが，同一組織や似通った目的を持つグループなど作業内容が似通っている場合には，一覧表に表示すべき機能数はそれほど多くないことが予想される．著者らは，ソフトウェアが提供する機能のうち，どのくらいの機能が実行されるかを調べるために，同一のドキュメントを共同で作成中の 3 人のユーザの機能実行履歴を 2 カ月間収集した．その結果当該ソフトウェア使用経験が 3 年以上持つユーザの間でも総実行機能数が約 7%程度でしかないことがわかった [9]．

3.2 機能実行パターンの一覧

機能実行パターンの実行頻度一覧は，ユーザが存在に気づいている機能を実行するためにあらかじめ実行しておかなければならない機能，存在に気づいている機能と組合わせることでより粒度の大きい機能の発見を支援する．機能実行パターンの一覧では，ある機能をユーザに指定してもらい，その機能を含む機能実行パターンの一覧を提示する．ユーザは指定した機能とともに実行される機能を見ることができ，その中から前提となる機能や組合せると作業効率が高まる機能を見つけることができる．

ユーザに提示する機能実行パターンのうち，ユーザが実行したことがある機能実行パターンの表示，非表示を切替えることができる．

一覧の各行は 1 つの機能実行パターンに対応し，表の各列には以下の項目を含み，機能実行パターンの各要素以外の項目で昇順，降順に並べ替えることができる．

- 機能実行パターンの各要素（$P_j(2 \leq j \leq m)$）
  機能実行パターンを構成する各機能．ユーザが指定した機能を含む機能実行パターンをユーザが指定した機能を中心として図 3のように KWIC(Key Word In Context) 形式で縦に並べる．単一機能の一覧と同様に，各機能にはその機能を実行するための選択経路が含まれる．ユーザは選択経路の表示，非表示を切替えることができる．

- 機能実行パターンの要素数（i）

- 他ユーザの実行割合（$count(f_i \geq 1)/$ t）機能実行パターンを 1 度以上実行したことがある自分以外のユーザの割合

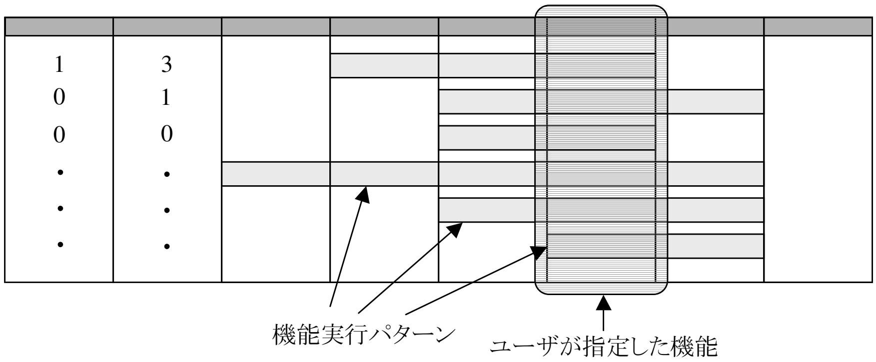

図 3: 機能実行パターンの一覧の表示形式

- 全ユーザの実行頻度（$\Sigma F_{j,k}$）全ユーザの機能実行履歴における機能実行パターンの出現頻度
- 各ユーザの実行頻度（$f_{j,k,u}$）各ユーザの履歴における機能実行パターンの出現頻度とその割合

ユーザは，ユーザが指定した機能を中心として，その機能を含む機能実行パターンから，その機能がどのようなコンテキストでどのように組合わせて使用されるかがわかる．例えば，ワードプロセッサや組版ソフトウェアで，目次を作成するためには章や節をなんらかの形であらかじめ設定しておく必要がある．「目次作成」機能を含む機能実行パターンに「章，節の設定」が含まれていれば，「章，節の設定」が「目次作成」の前提機能であることをユーザが推測できる場合がある．目次作成のように前提機能の実行頻度が高いものは，機能実行パターンに多く含まれ，ユーザが発見できる可能性が高い．逆に機能実行パターンに様々な機能が偏りなく含まれていれば，指定した機能の前提機能や組合せ機能をソフトウェアが提供していない可能性が高い．

前後に実行される機能と指定した機能をキーワードとして組合わせてオンラインヘルプを引けば，指定した機能だけをキーワードとするときよりも自分が必要とする使い方を知ることができる可能性が高まる．単一機能の一覧表と同様に機能名だけからある機能の前提機能や組合せると作業効率が高まる機能を発見できる可能性が高まる．

他ユーザの実行割合から，複数のユーザが同一の機能実行パターンを実行

していることがわかれば，そのパターンは一まとまりの作業手順であると考えることもできる．ユーザは，これらの機能実行パターンから組合せ機能や前提機能を知ることができる．特に，パターン長が大きい場合には，複数のユーザに共通するマクロのようなものであると考えられる．

4 試作システム

対象ソフトウェアを Microsoft Office 2000 としてシステムを試作した．機能実行履歴収集部，機能実行パターン提示部は Office2000 のプラグイン（add-in）モジュールとして実装した．機能実行履歴送・受信部は Java Servlet として実装され，web サーバに常駐し，ユーザの計算機と機能実行履歴サーバは http(hyper text transfer protocol) で通信する．

機能実行履歴収集部は，ユーザの Office 2000 に対するメニュー選択，ショートカットボタン押下，ショートカットキーの打鍵を監視し，日時，対応する GUI コンポーネントのキャプションを機能実行履歴として記録する．機能実行履歴はリアルタイムに機能実行履歴サーバに送出される．

機能実行履歴解析部は蓄積された機能実行履歴が一定量増加すると機能実行履歴に含まれるパターンを抽出し，蓄積しておく．試作システムでは機能実行履歴が 100 履歴増える度にパターンを抽出し，ユーザの要求に応えて蓄積したパターンを送信する．

単一機能の一覧，及び，機能実行パターンの一覧のスクリーンショットを図 4，5に示す．提示部は以下の機能を持つ．

- 一覧表: 各行に単一機能，機能実行パターンを表示する．各列には 3. で述べた項目を持つ．
- 最低出現頻度: 表示する単一機能，機能実行パターンの最低頻度を設定し，それ以上の出現頻度の機能，パターンを表示する．
- ユーザの選択: ユーザを選択することにより，自分がどのユーザであるかを指定する．
- メニュー階層の表示: 各機能，ルートメニューからの選択経路を示す文字列の表示，非表示を切替える．

図 4: 機能実行履歴提示部の単一機能の実行頻度の一覧表示

- 選択したユーザのパターン除去: 選択したユーザがすでに実行したことがある機能実行パターンの表示，非表示を切替える．

5　評価実験

5.1　概要

提案方式の有効性を確認することを目的として，4. で紹介した試作システムを用いた実験を行った．被験者は奈良先端科学技術大学院大学の学生 5 名である．被験者は Microsoft Word 2000 を用いてタスクを実行する．Microsoft Word 2000 は現在広く利用されているワープロソフトの一つである．また，1. で述べたとおり，多機能ではあるが，ユーザが実際に利用している機能はその一部に過ぎない典型的なソフトウェアである．なお，各被験者の Microsoft Word の使用年数，主な作成ドキュメントを表 3に示す．

被験者に与えられたタスクは，プレーンテキストをある形式の文書に整形することである．タスクの詳細は 5.2 で述べるが，タスクの作業効率を高める 5 つの機能がある．これらの機能は被験者には事前に知らせていない．5 つの機能の利用知識が，提案方式により獲得され，作業効率が高まるか評価す

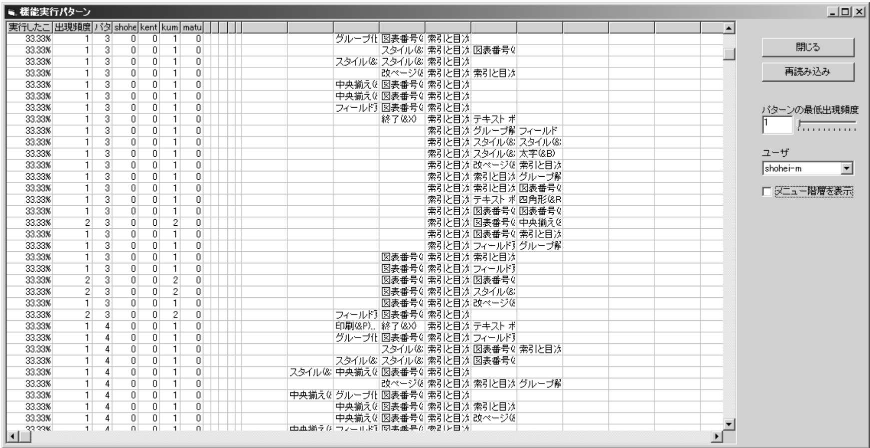

図 5: 機能実行履歴分析部の実行機能パターンの実行頻度の一覧表示

る．実験終了時に，各人の機能実行履歴を参照しながら，被験者にアンケートとインタビューを行い，タスクを効率よく遂行できる 5 つの機能について，既知であったか，実験中に利用知識を獲得したか，タスク遂行のため利用したか，を確認した．

5.2 タスク

被験者に与えるタスクは共通で，Microsoft Word 2000 を用いて（ファイルとして格納されている）プレーンテキストをある形式の文書に整形する作業である．被験者には，プレーンテキストのハードコピー，整形済み文書（できあがり見本）のハードコピーが渡される．整形すべき部分やその具体的内容は，できあがり見本のハードコピーに指示として書き込まれている．文書のできあがり時のサイズは A4 判 18 ページ，日本語で約 13,000 文字，7 章 11 節からなる．表を含み，図は含まない．

整形作業は次の 4 つの作業（サブタスク）T1 ～ T4 から構成されている．

T1： 目次の作成…章，及び，節見出しとページ番号の一覧を作成する．

T2： 章，及び，節見出しのフォント変更（17ヵ所）…文書全体を通じて，章と節の見出し部分のフォントを「MS ゴシック」に変更する（本文のフォントは，MS 明朝）．

T3：表の作成（3ヵ所）...タブで区切られた文字列を指示されたとおり罫線で囲み表を作成する．

T4：改ページ（14ヵ所）

Microsoft Word 2000は，これら4つのサブタスクT1 ~ T4を簡単に遂行することのできる次の5つの機能F1a, F1b, F2 ~ F4を提供している．F1aはF1bの前提機能である．また，F1aとF2を組合せることにより，より短い時間でタスクT2を実行できる．また，これらの機能の使用によりタスクとして与えた文書の整形で省ける操作の概数を表2に示す．F1aはF1bの前提機能，及び，F2の組合わせ機能であるため，F1a単体では操作数を減らすことはできない．表中の操作の単位は，打鍵，及び，メニュー選択の回数である．

F1a:「スタイル」...文書中に目次用の見出しを設定する．

F1b：「索引と目次」...文書中の見出しの一覧を，指定した場所に挿入する．ただし，機能F1aで見出しを設定しておく必要がある．

F2：「スタイル定義」...文書内の特定の文字列に対して，フォント，フォントサイズ，行間隔，文字配置等を一括して変更する．機能F1aと併用することにより，見出し部分のフォント，フォントサイズを再定義できる．

F3：「文字列を表にする」...タブなどで区切られた文字列を表に一括変換する．

F4：「改ページ」...ページを改める．

なお，被験者は，これらサブタスクT1 ~ T4と機能F1a ~ F4の存在を知らされていない．

5.3 実験手順

被験者にタスク，試作システムの使用方法を説明する．なお，被験者に対しては，次のような指示が出される．

- Microsoft Word 2000の既知の機能をできるだけ利用し，効率よくタスクを実行するように．

表 2: 実験に用いたタスクにおいて各機能により省ける操作の概数

機能名	使用時	非使用時
F1a: スタイル	18 操作	—
F1b: 索引と目次	20 操作	1500 打鍵
F2: スタイル定義	1 操作	51 操作
F3: 文字列を表にする	80 操作	300 操作
F4: 改ページ	8 操作	70 打鍵

- 必要があればオンラインヘルプを参照しても構わない．
- 実行が困難と思われる作業は後回しにしても構わない．

実験は次の三つのフェーズで構成されている．

フェーズ 1（p1）: 約 20 分間個別にタスクを実行してもらい，機能実行履歴を収集する．

フェーズ 2（p2）: 被験者に，試作システムによる単一機能，機能実行パターンの一覧表を閲覧してもらう．タスク実行に役に立ちそうな機能の利用知識の獲得を試みてもらう．閲覧時間の制約は特に設けない．

フェーズ 3（p3）: 被験者にタスク実行を再開してもらい，その終了まで作業を続けてもらう．なお，必要であれば，一覧表を引き続き閲覧することができる．

5.4 結果

被験者全員がタスクを終了させることができた．試作システムの機能実行履歴収集による計算機への負荷は非常に小さく，被験者のタスク終了までの時間への影響はない．

被験者ごとのフェーズ 1 とフェーズ 3 における作業時間，実行機能数，機能実行回数を表 4に示す．なお，機能実行履歴を被験者（ユーザ）間で共有するフェーズ 2 の所要時間は被験者毎に異なるが，おおよそ 5 分から 10 分であった．また，4 つのサブタスクの遂行に役立つ 5 つの機能について，各被

表 3: 被験者の Word 使用経験

被験者	作成ドキュメント	使用年数
S1	手紙，案内状	1 年
S2	レポート，案内状	2 年
S3	レポート，卒論	2 年
S4	レポート，論文 2	2 年
S5	レポート，案内状，論文 1	3 年

表 4: 被験者の作業時間，実行機能数，機能実行回数

	作業時間（分）			実行機能数			機能実行回数		
被験者	p1	p3	合計	p1	p3	合計	p1	p3	合計
S1	23	76	99	18	20	27	74	106	180
S2	23	22	45	17	15	26	91	29	120
S3	23	58	81	19	22	34	69	128	197
S4	23	48	71	15	17	28	84	50	134
S5	23	13	36	16	5	21	76	18	94

験者がどのフェーズでその利用知識を獲得し，タスク遂行に利用したかを図6に示す．図 6において，□印は利用知識の獲得を，△印は機能の実行を，それぞれ表す．△印のみの機能は，被験者にとってその利用知識が既知であったことを表す．また，□印も△印もない機能は，利用知識の獲得もタスク遂行のための実行もなされなかったことを表す．

サブタスク遂行に役立つ機能はのべ 25（5 × 5）個で，そのうち，被験者にとって利用知識が既知であった機能はのべ 9 個，実験中に利用知識が獲得されなかった機能はのべ 6 個であった．実験中に利用知識が獲得された機能はのべ 12 個で，そのうち，他の被験者の機能実行履歴を参照することで利用知識が獲得された機能はのべ 11 個，オンラインヘルプで利用知識が獲得された機能は 1 個のみであった．なお，試作システムを用いて他の被験者の機能実行履歴を参照することで利用知識が獲得された機能のうちの 2 つは，サブタスク遂行に役立つと予め想定した機能とは別の機能であった．

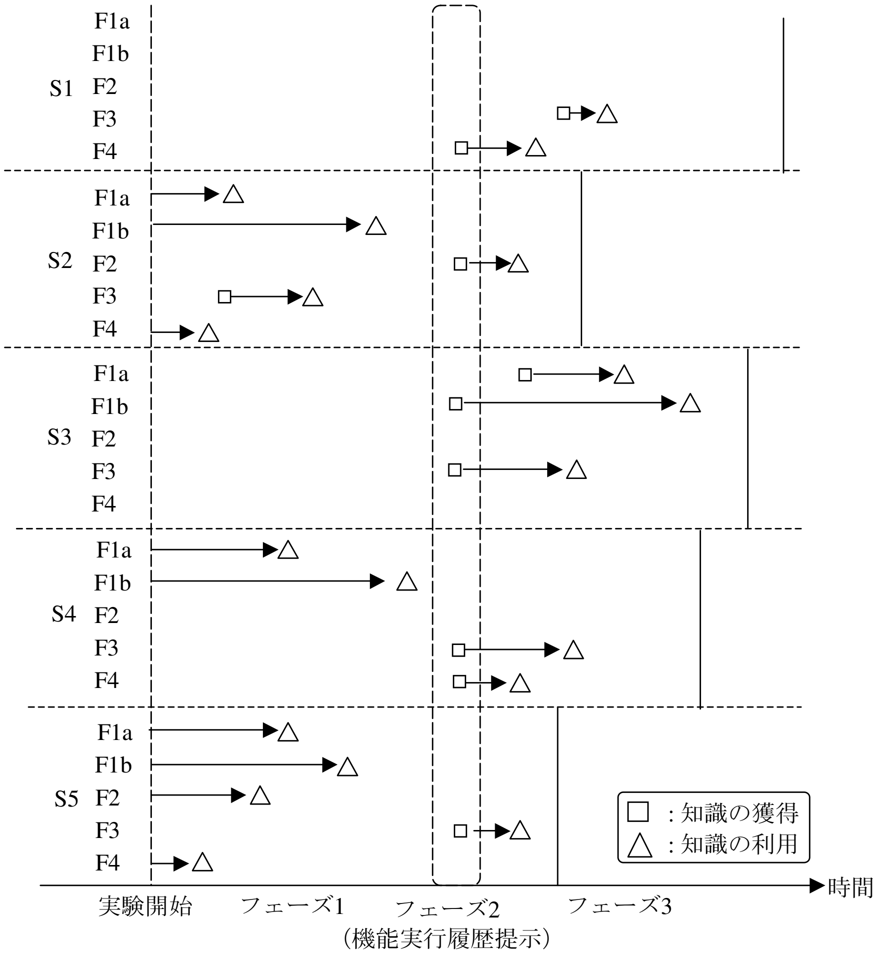

図 6: 各被験者の利用知識の獲得時期

表 4と図 6の情報，更に，実験後に被験者に対して行ったアンケートとインタビューの結果に基づいて，個々の被験者の特徴を簡単にまとめると次のようになる．

被験者 **S1**： 5 つの機能 F1a ~ F4 いずれの存在も知らなかった．しかし，他の被験者の実行した単一機能，及び，機能実行パターンを参照することで，F3, F4 の利用知識を獲得し，フェーズ 3 においてサブタスク T3, T4 遂行のために実行している．機能 F1b の存在には気づいたが，機能 F1a がその前提機能であることに気づかなかったため，機能 F1b を実行することはできなかった．機能 F3，F4 の存在には気づかなかった．フェーズ 3 の作業時間は

76 分であり 5 名中最も大きい．

被験者 **S2**： 3 つの機能 F1a, F1b，F4 の存在を知っていた．また，F1a が F1b の前提機能であることを知っていた．機能 F2 についてはその存在は想像していたが，他の被験者の機能実行パターンを参照することで，機能 F2 が F1a の組合せ機能であることを知り，サブタスク T2 のために使用している．機能 F3 については，フェーズ 1 においてオンラインヘルプを参照することで，その利用知識を獲得し，サブタスク T3 遂行のために実行している．5 つの機能 F1a ~ F4 を全て利用してサブタスクを実施した被験者の一人で，結果として，フェーズ 3 の作業時間は 22 分と 5 名中 2 番目に小さい．

被験者 **S3**： 被験者 S1 と同様に F1a ~ F4 いずれの機能の存在も知らなかった．単一機能一覧により，フェーズ 2 において F1b の存在に気づき，フェーズ 3 において機能実行パターンの一覧から F1a が F1b の前提機能であることを知り，サブタスク T1 のために使用している．機能 F2, F4 の存在には気づかなかった．また，予め想定した 4 つの機能 F1a ~ F4 とは別の「箇条書きと段落番号」機能の存在を，フェーズ 2 において単一機能一覧から知った．

被験者 **S4**： F1a と F1b の機能の存在，F1a が F1b の前提機能であることを知っており，フェーズ 1 でサブタスク T1 を遂行するために実行している．フェーズ 2 において，単一機能の一覧から機能 F3，F4 の存在に気づき，フェーズ 3 においてサブタスク T3, T4 のために実行している．

被験者 **S5**： 機能 F3 以外の機能の存在，F1a が F1b の前提機能であること，及び，F1a が F2 の組合せ機能であることを知っていた．フェーズ 2 において単一機能の一覧から機能 F3 の存在を知り，フェーズ 3 においてサブタスク T3 のために実行している．5 つの機能 F1a ~ F4 を全て利用してサブタスクを実施した被験者の一人で，結果として，フェーズ 3 の作業時間は 13 分と 5 名中最も小さく，被験者 S1 の約 5 分の 1 になっている．

6 考察

5. で述べた評価実験は，被験者数も少なく，その結果を一般化することは適当でないかもしれない．しかし，得られた結果のいくつかは，ソフトウェア利用知識の共有，及び，共有による作業効率の向上に提案方式が役立つ可

能性のあることを示唆していると考える．例えば，次の点である．

- 単一機能の一覧は初心者（被験者 S1）を含め，すべての被験者に存在に気づいていない機能の存在に気づくことを支援した．また，単一機能の一覧を閲覧する時間は 10 分程度であり，比較的短い時間でその後の作業効率を高くすることができた．単一機能の一覧は，小さいコストで初心者から熟練者までの作業効率の向上に寄与する可能性がある．

- 前提機能や組合せ機能を知らないことにより実行できなかった機能があった 4 人の被験者のうち，2 人の被験者が機能実行パターンの一覧により，既知の機能の前提機能，組合わせ機能の存在に気づき，既知の機能を実行することができた．機能実行パターンにより，存在には気づいているが，前提機能を知らないため実行できない機能や，組合わせることで作業効率をより高めることができる機能を知ることができる可能性がある．

- 提案方式の有効性をより明確に評価するために（被験者には知らされていなかったが）タスクには 4 つのサブタスクとそれらを簡単に行うための 5 つの機能が予め設定（想定）されていた．しかし，それら 5 つの機能以外の機能についても，その利用知識を獲得した被験者が 1 名いた．このことは，タスク中に予め設定されていた機能のみが，提案方式による利用知識の獲得に特に適していたわけではないことを示している．また，提案方式を実際に利用する場合に当てはめてみると，ユーザに利用知識を獲得させる機能を予め明らかにしタスクを設定する，という必要が必ずしもないことになる．提案方式は，適用範囲が広く，実施コストも比較的小さい可能性がある．

- 機能実行履歴には，実行されたメニュー項目名だけでなく，メニュー階層内での位置が分かるよう，ルートメニューから実行されたメニュー項目名までの選択経路が付加されている．こうした情報は，ユーザがオンラインヘルプ等で機能の詳細説明を検索する際の非常に適切なキーワードとなる．もちろん，ユーザがその機能を試用しようと思えば，示された選択経路に従って実際にメニュー項目を選択するだけでよい．オンラインヘルプ等の既存のユーザ支援機構と提案方式をうまく連携させることで，利用知識の獲得コストを大幅に小さくできる可能性がある．

7　関連研究

ソフトウェアを利用した作業効率の向上を目的として，ユーザの操作履歴を利用した適応／予測型インタフェースを持つ多くのシステムが提案されている [8][15][16]．しかし，これらのシステムはユーザ自身の操作履歴を利用するため，ユーザが存在に気づいていない機能を提示，予測することはできない．

機能の存在や使い方（機能の利用知識）を獲得するコストを小さくするための支援の一つに知的ヘルプシステム [5] [7][17] がある．知的ヘルプシステムではソフトウェア開発者がユーザモデルをあらかじめ構築しておき，それに基づき，ユーザが気づいてないと推測される利用知識を適切なタイミングで提示することにより，ユーザの利用知識獲得コストを小さくしている．ただし，ソフトウェアに依存するユーザモデルの構築が必要となるため，膨大な機能を提供するソフトウェアにおいて効果的なユーザモデルを構築することは容易ではない．提案方式では，ユーザモデルを構築することなく，ユーザ間で機能の利用知識を交換することを支援することにより，作業効率を向上することが可能である．

ソフトウェアの利用知識を対象としたものではないが，個人の暗黙知を有形化し，グループで共有するシステムとしては Advice/Help on Demand が知られている [13]．業務ノウハウを文書化しグループ内での参照を可能にしている．但し，業務ノウハウが自然語で記述されているため，本提案方式のように，ユーザの本来の作業を妨げずに，収集，分析することは容易ではない．

本提案手法と同様に，ソフトウェアユーザの行動やソフトウェアを利用した業務内容を記述する手段として，ソフトウェアの実行（操作）履歴を用いる研究は数多く報告されている [2][4]．例えば，森らは，X-window におけるマウス操作や打鍵などの詳細な操作履歴を収集，分析するツールを提案している [10]．彼らのシステムを用いれば，操作履歴に基づいて，ユーザ操作の再現，正規表現に似た形式での操作履歴の分析が可能である．また，安部田らは，個人情報管理システム（PIM）の操作履歴を収集し，企業における業務知識を体系化するシステムを提案している [1]．

ソフトウェアの実行（操作）履歴をネットワーク経由で収集するシステムはいくつか提案されている．Hilbert らは，ソフトウェア設計者の意図する操

作系列と異なる操作をユーザが行うと，その詳細を電子メイルで設計者に通知するシステムを開発している [6]．彼らのシステムを用いれば，ソフトウェア操作に対する設計者とユーザの認識のずれを知ることが出来，ソフトウェア設計に役立てることができる．但し，ユーザの操作と比較するための操作系列を設計者が予めシステムに登録しておく必要がある．本提案方式のように，実際の機能実行履歴の中から利用知識を抽出することはできず，ユーザ間での利用知識の共有を支援する機能も有していない．

8　おわりに

ソフトウェアが提供する機能の利用知識獲得コストを小さくすることを目的として，機能実行履歴の提示方式を提案し，提案方式に基づく試作システムと評価実験の結果について述べた．提案方式では，ユーザがその存在に気づいていない機能，及び，ある機能と組合せて実行する機能やある機能の前提となる機能の発見に適した形式でユーザに提示する．評価実験では，被験者 5 名全員が今まで知らなかった機能を 1 つ以上発見することができ，そのうち 2 名がある機能の前提機能や組合わせ機能を発見することが確認された．

本稿で述べた評価実験では，被験者全員が同一のタスクを実施する際の知識獲得の様子を調べたものである．現実に提案方式を使用する際にはユーザが必ずしも同一のタスクを実施するとは限らない．現実の状況における実験的評価により提案方式の適用範囲を確認することが今後の課題である．

参考文献

[1] 安部田 章: ユーザの操作履歴から業務知識を抽出する個人情報管理システム，情報処理学会 技術報告 GW23-2, pp.7–12, 1997.

[2] 赤池 英夫，角田 博保: X-Window 上の利用者行動分析システム, 情報処理学会論文誌 Vol. 33, No. 5, pp. 736-745, 1992.

[3] J. M. Carroll and M. B. Rosson : Paradox of the Active User, Interfacing Thought: Cognitive Aspects of Human-computer interaction, MIT Press, 1987.

[4] 海老名 毅，伊藤 昭: X アプリケーションにおける可読性のある操作履歴の取得について, 電子情報通信学会 技術報告 HC94-87, pp. 1-7, 1995.

[5] M. Hecking: How to Use Plan Recognition to Improve the Abilities of the Intelligent Help System SINIX Consultant, Proc. of INTERACT, pp. 657-662, 1987.

[6] D. M. Hilbert and D. F. Redmiles: An approach to large-scale collection of application usage data over the Internet, Proc. of 20th International Conference on Software Engineering, pp.136-145, 1998.

[7] 伊藤 昭，海老名 毅，熊本 忠彦: 対話型計算機利用支援におけるユーザ質問の分類と支援回答戦略，電子情報通信学会論文誌，vol. J77-D-II No.7 pp.1319–1328, 1994

[8] Masui. T. and Nakayama. K. : Repeat and Predict — Two Keys to Efficient Text Editing, proceedings of the ACM Conference on Human Factors in Computing Systems, Addison-Wesley, pp. 118-123, 1994.

[9] 松本 健一，森崎 修司: " フィールドテストにおけるソフトウェア操作履歴の収集と分析の支援"，平成 11 年度高度情報化支援ソフトウェアシーズ育成事業報告書, < http://tori.aist-nara.ac.jp/usageLog/ >，2000.

[10] 森 孝弘，西田 知博，斎藤 明紀，都倉 信樹: 大量の GUI 操作履歴を分析するための走査・再生ツール，情報処理学会研究報告，HI-69-1，pp. 1–8，1996.

[11] 森崎 修司，門田 暁人，松本 健一，井上 克朗，鳥居 宏次: 機能実行履歴を用いたソフトウェア利用知識の共有，情報処理学会論文誌（投稿中）.

[12] Shuuji Morisaki, Akito Monden, Ken-ichi Matsumoto, Katsuro Inoue, and Koji Torii: Two-level User Interface Evolution by Sharing Usage Logs, Proceedings of International Workshop on Software Evolution, pp. 70–73, 1999.

[13] 中山 康子，真鍋 俊彦，竹林 洋一: 知識情報共有システム (Advice/Help on Demand) の開発と実践:知識ベースとノウハウベースの構築，情報処理学会論文誌，vol.39, No.5，pp. 1186–1191，1998.

[14] D. Reisberg: Cognition，New York, NY: W. W. Norton & Company, 1997.

[15] Sears A. and Shneiderman B. : Split Menus: Effectively Using Selection Frequency to Organize Menus, ACM Transaction on Computer-Human Interaction, Vol. 1, No. 1, pp. 27-51, 1994.

[16] 杉浦 淳，古関 義幸: 例示プログラミングにおけるマクロ定義の簡略化，日本ソフトウェア科学会 インタラクティブシステムとソフトウェア IV，近代科学社，pp. 101-110, 1996.

[17] 高田 光男，西野 順二，小高 知宏，小倉 久和: UNIX 高機能シェルの行編集機能に対する適応型ヒューマンインタフェースの構築とその評価，情報処理学会 論文誌，vol. 38, No. 10，pp.1919–1927，1997.