ToDoリストを利用したノウハウ探索支援システム「Shumu」

慶應義塾大学 経済学部
前嶋陽一

指導教員
慶應義塾大学 環境情報学部
村井 純
徳田 英幸
中村 修
武田 圭史
楠本 博之
高汐 一紀
植原 啓介
三次 仁
重近 範行
Rodney D. Van Meter III
中澤 仁

平成22年2月13日

概 要

近年の WWW(World Wide Web) の進歩により多くのノウハウへ容易にアクセスできるようになったが、(1) 外的ノウハウ探索未発生による機会損失問題、(2) 未言語化ノウハウの探索問題、(3) 欲求認識における背景知識不足問題といった問題をはらんでいる。本研究ではノウハウ探索の促進と支援を行うシステム「Shumu」を作り、人のノウハウの探索を支援することを目指す。

Shumu はユーザの抱える「タスク」を把握し、ユーザが検索行動を起こさなくてもリアルタイムに適切なノウハウを検索しユーザに表示する。多くのユーザにとって理解しやすい「ToDo リスト」のインターフェイスを実装し、入力された ToDo をタスクの把握に活用する。また過去の ToDo データはユーザがどのように行動してきたかを表し、「行動ログ」として解釈する。

本研究では、「人は他者の行動ログをみることで、ノウハウを模倣することが出来る」と考える。今まで見る事が出来なかった他者の行動ログを検索可能にし適切なタイミングで表示することで、ノウハウの模倣を促す。

Shumu ではユーザの抱えている「タスク」にとって参考になる行動ログを検索するために、ユーザ同士のつながりであるソーシャルグラフを用いる。「同じクラスタ（ユーザの社会的距離の近さ）に属するユーザの行動ログは行動の背景が近いことが多く、社会的距離の遠いユーザの行動ログより参考になる」と考えるためである。

ToDo データは上位の ToDo が下位の ToDo の完了によって表される「is-achieved-by 関係」で構築されたツリー構造として入力する。そのツリー構造の持つ情報も行動計画中のタスクと行動ログのマッチングに利用する。ToDo のマッチング処理はユーザが近隣のノードに問い合わせ、問い合わせを受けたノードは所有する過去の ToDo から関連しそうなデータを検索し、返答する。

Shumu をインスタントメッセージングプラットフォームである「wija」のプラグインとして実装することで、通信に関わる機能を wija に頼る。他ユーザの過去 ToDo データからだけではノウハウを掴むことが出来ない場合は、「wija」に備わった多種のコミュニケーション手段で直接「尋ねる」ことをシームレスに行うことが出来る。

本稿では Shumu のアプローチを検証するためにユーザに課題を与え、実際に Shumu のデモアプリケーションで行動計画をしてもらい、その様子を観察した。その結果、ToDo リストのユーザインターフェイスであれば多くのユーザにとって理解しやすいとは結論できず、インターフェイスの理解を促す更なる工夫が求められる事が分かった。経験が多いユーザほど ToDo をより具体的に分解している事がわかり、ToDo リストを経験の一部として使う事が出来ることが分かった。

本研究によって Shumu によってノウハウの探索支援の可能性が示唆されたが、ユーザに ToDo リストを適切に入力してもらうには更なる改善が求められることも分かった。今後はユーザインターフェイスの改善し、Shumu を広く利用してもらって検証することが求められる。また、Q&A システムなどの他のノウハウ探索システムとの組み合わせや、マッチングにおける知識処理技術の導入などの改善が行える。

Abstract

ToDo-based Knowhow Search Supporting System "Shumu"

With the development of the WWW (World Wide Web), it become easier to access much knowhow. However, it has three problems; (1)opportunity loss on the knowhow-seeking, (2)difficulty on the unverbalized knowhow-seeking, (3)the lack of background knowledge on the task definition. This study aims to make knowhow-seeking better by constructing the knowhow search supporting system "Shumu".

Shumu recognize the user's task, and show appropriate knowhow automatically. Shumu has the ToDo-list like user interface which is familiar to people. Shumu apply ToDo-list information for recognition of user's tasks, and interpret ToDo-list which is accomplished as user's action logs.

This study makes hypothesis that user can imitate other people's action by watching his or her action logs. Shumu makes it easier to watch other people's action logs, and helps users to imitate other people's action.

Shumu uses the user's social graph to search other people's action logs which is informative for the user. A social graph is social structure which contains user nodes and user relation edges. It is seemed that users belong same social cluster have same social contexts, the smaller social distance between two users is, the more informative knowhow is.

Shumu treat the ToDo-lists as tree sturcuture. The relation between parent ToDo and child ToDos is called "is-acheved-by relationships", and this stracutural information is used for ToDo-matching. ToDo-matching is processed as distributed searching. A user who search knowhow query other users who is near on the social distance, and the user who received the query searches in own ToDo-lists, and if the relative action logs are found, the query receiver returns relative action logs to the query sender.

Shumu is implemented as a plug-in of "wija". Wija is an instant messaging platform, and the communication function of shumu depends on wija's function.

In this paper, Shumu was verified through an experiment. Some users used Shumu for planning tasks, and I observe how Shumu was used. As a result, the ToDo-list user interface is not comprehensible and it is found that further improvements are required on the user interface. The experiment also shows that users who have much experience of the task tend to plan more concrete task-breakdown than users who have less experience.

This research suggests that Shumu helps knowhow-searching, however further improvements are required to collect correct ToDo-lists. It is an issues in the future that the ToDo-list user interface should be improved, and Shumu should be used by many people as the experiment. In addition, Shumu should be integrated other knowhow sharing system such Q&A system, or knowhow of the knowledge processing should be applied to ToDo-matching.

目 次

図 目 次

表 目 次

第1章　序論

1.1　はじめに

人は行動するときに、「行動に関する具体的な方法論的知識」である「ノウハウ」に基づいて行動する。これまでに経験した行動をとるとき、人は自らの記憶のなかからノウハウを探索しそれを応用して行動している。今までに経験した事がない行動をとるとき、その行動のためのノウハウをもつ人に「聞く」か、図書館やWeb検索でどのように行動するかを探索する。ノウハウの存否は行動の成否や効率性に大きく関わるため、アクセスできるノウハウの量や範囲によって人々の行動は制限されていると言える。

近年、Web(World Wide Web)の拡大とWeb検索エンジンの進化に伴い、人々は多くのノウハウに容易にアクセスできるようになった。Webを使えば、いままで図書館で何時間もかかっていた調べていたようなことも、数分で調べる事が可能になった。Webはノウハウ探索において、獲得できるレパートリーの増加や探索時間の短縮といった劇的な変化を起こした。そしてそれは、人々がノウハウを手に入れることを強力に推進し、行動の成功率や効率性を上げ、行動範囲を広げる事に貢献した。

そしてWebは人々の情報探索の中心的な方法になりつつある。みずほ情報総研が2007年4月に行った「ビジネスシーンにおける情報探索行動に関するアンケート調査」では、「さがす方法においてITを使わない割合」を訪ねたところ、その割合の平均値は35 %であった。つまり平均的なビジネスマンは65 %以上をITを使って「探す」という行動を行っている。さらにITを用いる人の91.78 %「検索サイト」を用いると答えている。いまやWeb検索エンジンを用いた情報の探索は探索行動において最も高い割合を占めるまでになった。[14]

しかし、ノウハウ探索をWebに依存する事で新たな問題が生まれつつある。Web検索に依存し、人に聞くという探索行動をとらなかったために、ノウハウ探索にかける時間がかかりすぎてしまう問題が多く報告されている。[3] [2] [1] これらのブログでは、職場で先輩社員に聞けばすぐに分かる問題に対して、Web検索のみによって検索を試み、答えを探し出せない新入社員の姿が紹介されている。Web検索を情報探索の主要な方法として行う世代において、行動を起こす時のノウハウ探索方法としてWeb検索に依存しすぎることの問題が浮かび上がってきている。

Web検索技術が進歩し、多くの人がWeb検索を使いこなすと同時に依存している時代において、本研究はノウハウ探索においてさらなる改善を目指す。

1.2 目的

本研究ではノウハウの探索方法としてWeb検索が抱える問題を分析し、人に「聞く」という探索方法も含めノウハウ探索行動全般を支援するシステムを構築する事で、人々がアクセスできるノウハウの範囲を拡大する事を目指す。そのために、人がノウハウ探索においてどのように行動するかモデルを構築し、どのような探索コストが発生しているのか明らかにする。そして、特に人に「聞く」という行動がWeb検索によるノウハウ探索の欠点を補うことを示し、そのコストを低減するシステムを構築し、それによってノウハウ探索行動の改善を目指す。

1.3 本研究で扱う問題

本研究ではノウハウ探索を行うユーザを支援するシステムを構築する事を目的としている。そこで現状のノウハウ探索の持つ問題、特にWeb検索が情報探索において占める割合が高くなってからの問題を分析したい。特にWebとWeb検索が主流になっている近年の問題を指摘する。

1.3.1 外的ノウハウ探索未発生による機会損失問題

一般的なユーザは、ほとんどの行動において内的情報探索しか行わず、外的情報探索は行わない。外的情報探索が行われない理由は、ほとんどの習慣的な行動について、ユーザにとって外的情報探索によって効用増加が起こらないとユーザ自身が予想するためである。2.2.3 だが、本人が行おうとしている行動について、本人がもっているノウハウだけで十分効率的に行えるのか、もしくはまだ持っていないノウハウを探索することでさらに効率的に行えるのか、本人は知ることができない。「Web検索」はユーザが探索行動を起こしたときだけ情報を提供する「プル型」のメディアである。ユーザが主体的に情報探索を行わなくてもメディアの側から情報を提供する「プッシュ型」のメディアではない。プル型のメディアにおいて「無知の知」を得ることは難しいと言える。

ユーザは主観的に「自分の知識量は十分か」を推定することができるのみであり、ユーザが知らないノウハウが存在しても、ユーザは自らが知らないことを知ることはできない。すると新しく実行した場合に得られる「追加で情報を手に入れたときに増加する効用の見込み」が過小評価され、「効用増加期待値」は下がる。そのため、ユーザは外的ノウハウ探索を行わずに内的ノウハウ探索のみで行動を計画することが多く、より良いノウハウによって行動を効率化する機会を逃してしまっていると考えられる。本研究ではこの問題を「外的ノウハウ探索未発生による機会損失問題」と呼ぶ。

問題 1 (外的ノウハウ探索未発生による機会損失問題)**.** ユーザが自らの記憶から内的ノウハウ探索だけしか行わず、外的ノウハウ探索をしない事で、ユーザが記憶しているノウハウより優れているものを知る機会を失われているという問題。

1.3.2 未言語化ノウハウの探索問題

システムで探索を行うとき、検索対象となるものは、システムが検索可能な形で表現されていなければならない。コンピュータシステムでは文字のみならず動画や画像なども検索対象に含めることが可能ではあるが、これらの「システムで検索可能な形として知識を表現すること」を本研究では「言語化」と呼ぶことにする。つまり、言語化された知識は検索対象となるが、言語化されていない知識はシステムでは検索できない。そして言語化されていない知識が存在することは示す事は出来ないが、多くの人に理解してもらえるだろう。「我々は語るより多くのことを知ることができる」のである。[10]

特に近年よく使われるようになった Web と Web 検索エンジンについては、言語化され Web 上に表現されたもののみが検索対象となる。Web 検索エンジンは Web 上のデータをクロールし、それらを高速に検索できる形に変換する「インデックス作成処理」を行っている。しかし、Web には言語化されたノウハウしか存在せず、言語化されていないノウハウは Web 検索エンジンの対象にはならない。この問題を「未言語化ノウハウの探索問題」と呼ぶことにする。

近年、Web 検索技術は進歩し、「正しい情報が見つかる主観的確率」は上昇し、また、ユーザにとっての「探索コスト予想値」が下がった。そのためユーザは Web と Web 検索エンジンの組み合わせによる探索方略を選ぶ傾向が強まってきている。Web 検索のみに頼ると、これらの未言語化情報の存在に気づかないで終わってしまう。

問題 2 (未言語化ノウハウの探索問題)**.** 言語化されていないノウハウは *Web* 検索などの既存の探索手段で探索できない。それにより言語化されていないノウハウと出会う機会が失われているという問題。

1.3.3 欲求認識における背景知識不足問題

背景や関連知識がないと問題が何であるか、問題の認識がうまく行えない、もしくは問題を探すための検索キー（検索キーワードや質問文など）を思いつかない可能性が高い。人は他者のノウハウから背景知識や関連情報を類推する能力を持つが、あまりに背景知識が不足していると誤った類推をしてしまう。

このような問題を「欲求認識における背景知識不足問題」と呼ぶ。

例えば、プログラミングの初心者が、プログラムが動かないと悩み、様々なサンプルコードを見て動かすための工夫をするが、実はプログラムの起動方法を間違っているために動かないというケースである。このケースではプログラムを起動するコマンドや、動く仕組みを理解していればコードの問題でないことは分かるのだが、初心者は知識が不足しているために真の問題に気付く事が出来ない。

問題 3 (欲求段階における背景知識不足問題)**.** ユーザの背景知識が不足していたために、ユーザが本来抱えていた目的には適合しないタスクを設定し、関係のない探索をしてしまうという問題。

1.4 論文の構成

本論文は 7 つの章からなる。第 2 章では本研究の背景となる探索行動モデルや他の研究分野における問題へのアプローチを説明する。第 3 章では本研究で構築する探索行動支援システム「Shumu」がどのように問題にアプローチしていくのか、その方法論を述べる。第 4 章では「Shumu」の具体的な設計について説明する。第 5 章では「Shumu」の実装について説明する。第 6 章で「Shumu」のアプローチの元になっている仮説について検証を試み、その結果を検証する。第 7 章において本研究をまとめ、今後の課題について述べる。

第2章　背景

2.1　ノウハウの定義

本研究で対象とするノウハウとは「何らかの作業に直接適用可能」な知識のことである。[9]

行動を行う際には、その行動をどのように行えばよいか、ノウハウが探索され、適用される。ノウハウは「実地の経験、問題解決の慣習、特定の解決法の限界への理解といった感覚を伴う」傾向がある。[9] そのため、理論では解法を説明できてない問題においても、ノウハウは存在しうる。そう言った意味でノウハウは理論を超越するものである。しかし同時に、ノウハウは特定の状況下、特定の固有領域においての知識が無意識に前提されていることが多いため「宣言的知識よりも汎用性に乏しい傾向」があるとも言える。[9]

2.2　探索行動モデル

2.2.1　行動ステップ

本研究では人間が行動を行うときの行動を３つのステップからなると分析する。

まずはじめに、「欲求の認識」というステップを踏む。これは「自らの現状における欠乏」を感じ、その行動をとるべきである理由を発見することである。そして「将来の望ましい状態を認識」し、理想状態までどのように到達するべきかという問題として認識する。[6]

次にユーザは「欲求の認識」で認識された「どのように理想状態に到達するか」という問題に対しての解を探索し、どのように行動するかという計画を立てる。このステップを「行動計画」と呼ぶ。行動計画はさらに「ノウハウ探索」と「ノウハウ適用」に分けられるが、その二つは並列的に行われる。

「ノウハウ探索」とは、過去に同じような問題を解決した事がないか自らの内的記憶からノウハウを探索する。そして適切なノウハウが記憶になかった場合は場合は人に聞く、Web で検索するなどの外的探索行動を別に行おうとする。詳細については 2.2.2にて説明する。

そして見つかったノウハウを問題に適用するために変形し、さらに適用可能かというステップを踏む事になる。このステップを「ノウハウ適用」と呼ぶ。

ユーザは行動計画を立てた後、計画に基づいて「行動」し、当初認識された欲求を満たす。このように物事が進んでいく。

この人間の行動モデルについて図示すると図 2.1のように表す事が出来る。

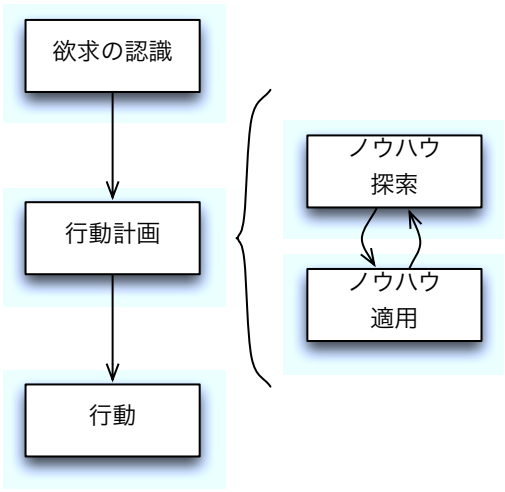


図 2.1: 行動モデル

2.2.2 ノウハウ探索モデル

「現状から目標状態までどのように到達するか」という問題において、具体的な道筋を示す情報を探すという行為が「ノウハウ探索」である。ノウハウ探索には内的ノウハウ探索と外的ノウハウ探索がある。

ノウハウ探索モデルは消費者行動理論における情報探索モデルを参考にした。[13] 消費者情報行動理論では、消費者がどのように探索行動をとるかを分析し、消費者の意思決定に情報がどう活かされているかを考察する。

内的情報探索とは、行動を起こす者が自らの記憶の中から該当する情報を探索することであり、無意識的に行われる。外的情報探索とは行動を起こす者が「他者に聞く」、「Webで検索する」、「図書館で調べる」といった外的情報源を探索し、アクセスすることで情報を得る行為である。外的情報探索はコストを伴うために、「外的情報探索をするべきか」という判断が行われ、する必要があると認識された場合のみ実施される。その様子を示すと図 2.2として表す事が出来る。

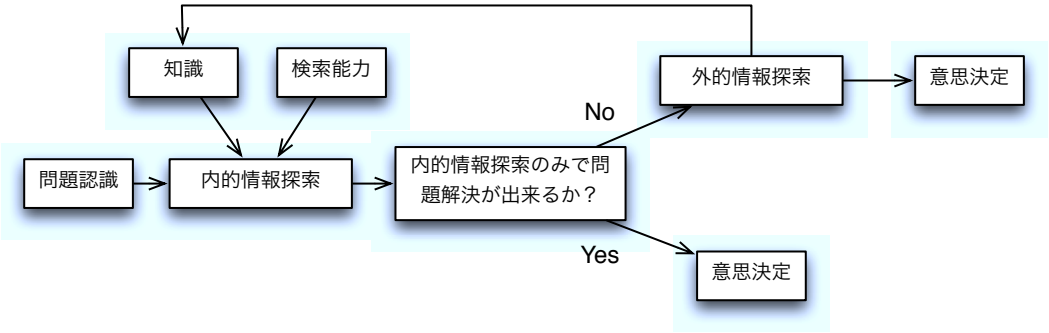


図 2.2: 情報探索モデル

図 2.2のモデルは情報としてあるが、ノウハウは消費者行動理論の指す「情報」の部分集合であり、「情報」を「ノウハウ」と置き換える事で、同じ構造でモデル化が可能であると考えられる。

2.2.3　外的ノウハウ探索方略の選択

「外的ノウハウ探索をどの情報源から行うか」という意思決定がどのように行われるかについて分析を試みる。

節 2.2.2では「外的情報探索を行うか否か」という意思決定のみを行っていた。本研究では、外的情報源がどのように選ばれるのか考察を加え、モデルの発展を試みる。

内的情報探索が終了すると、その情報で十分か否かが検討され、外的情報探索が必要と判断された場合のみ外的情報探索が行われる。外的情報探索ではどの情報源を選ぶべきか、ユーザは方略を決定しなくてはならない。

ユーザが経済合理的な判断を行う主体であると仮定すると、ユーザは各探索方略をベネフィットからコストをひいた正味の効用で比較し、もっとも効用が高まる選択をしていると考えられる。つまり探索方略選択の意思決定は「探索によって増加する効用の期待値」から「探索する事によって負担しなければならないコスト」を引いた値で各探索方略を比較する事で行われる。「探索によって増加する効用の期待値」は「追加で情報を手に入れたときに増加する効用の見込み」に「正しい情報が見つかる主観的確率」を乗じたもので表される。この値を「効用増加期待値」と呼ぶ。詳細について小節 2.2.4で説明する。「探索する事によって負担しなければならないコスト」は外的ノウハウ探索の各ステップ（図 2.5参照）で発生するコストの予想値の総和で表される。この値を「探索コスト予想値」と呼ぶ。そして意思決定に用いられる「効用増加期待値」から「探索コスト予想値」を引いた物を「探索方法評価点数」と呼ぶこととする。ユーザは限られた探索時間内で、「探索方法評価点数」が最も高い「探索方法」を選ぶ。

2.2.4　効用増加期待値

効用増加期待値は「追加で情報を手に入れたときに増加する効用の見込み」と「正しい情報が見つかる主観的確率」の積として表す事が出来る。それぞれについて説明する。

2.2.4.1　追加で情報を手に入れたときに増加する効用の見込み

ユーザはノウハウを手に入れる事で行動の成功確率を上げる、効率を上げるといった、自らにとって好ましい変化が発生する事を目指して探索している。そこで新しく有用なノウハウが見つかった場合、当然ユーザの効用は増加する事が予想できる。この増加する値の期待値が「追加で情報を手に入れたときに増加する効用の見込み」である。

ユーザにとって影響力の大きな行動においては、ノウハウによる効率の改善や成功確率の向上は大きな効用をもたらす。例えば、1 万円の仕事と 3 億円の仕事では、3 億円の仕事に関するノウハウ探索のほうが力が入る。また逆に損害リスクが大きな作業に関してはノウハウ獲得によってそのリスクを減らそうとするため、ノウハウ探索にかけるコストは高くなる傾向がある。

ユーザがノウハウを持てば持つほど、追加で得られる効用見込みは小さくなる。ユーザが特定の問題についてのノウハウを十分に所有している場合、追加でノウハウを手に入れたときに効率の改善やリスクの逓減はあまり見込めないが、あまりノウハウを持っていない分野においては少しのノウハウでも大きく効率の改善、リスクの低減が見込める。つまり、ユーザのノウハウ蓄積量が増えるにつれて効用は逓減していく。追加で得られる効用見込みはユーザのノウハウ所有量の増加に伴い減少していく。(図 2.3参照)

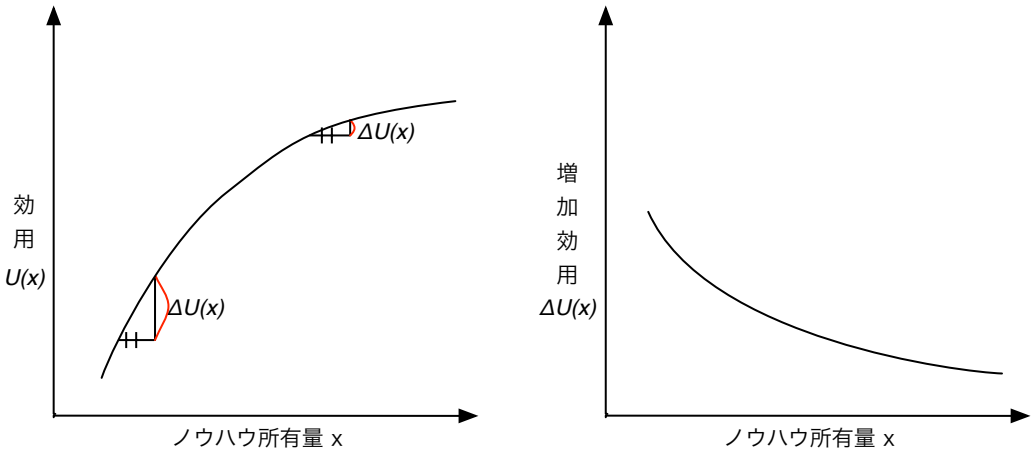


図 2.3: ノウハウ量に対して逓減する効用

追加で情報を手に入れたときに増加する効用の見込みは、与えられた問題とユーザのノウハウ所有量に依存するが、ノウハウ探索方略には影響されない。ノウハウ探索方略の選択時にどの探索方略を選ぶか、という選択には影響を及ぼさないが、外部ノウハウ探索を打ち切るという判断には影響を及ぼす。

2.2.4.2 正しい情報が見つかる主観的確率

ユーザがノウハウ探索活動を行う事によって、正しいノウハウが見つかるユーザの主観的な確率である。ユーザは過去にその探索方略を選択したときに正しいノウハウが見つかった経験から、探そうとしているノウハウがその探索方略によって見つかる確率を主観的に予想している。これは探索方略によって異なる「探索できる情報の範囲」や「情報源の信頼性」などの定性的なデータから、ユーザが無意識に算出している。この値は、節2.2.4.1で説明した「追加で情報を手に入れたときに増加する効用の見込み」と違い、探索方略ごとに異なる。

「正しい情報が見つかる主観的確率」はあるノウハウ探索方略を継続していると、徐々に減少していく。ユーザはそのノウハウ探索方略を続ければ続けるほど「探し続けても見つからないのではないか」という不安を持ち始める。探索活動を続ければ続けるほど、その情報源に求めている情報がない可能性が高いのではないかと感じるようになる。つまり一般的に、探索量が多くなれば多くなるほど、主観的発見確率は逓減すると言える。(図2.4)

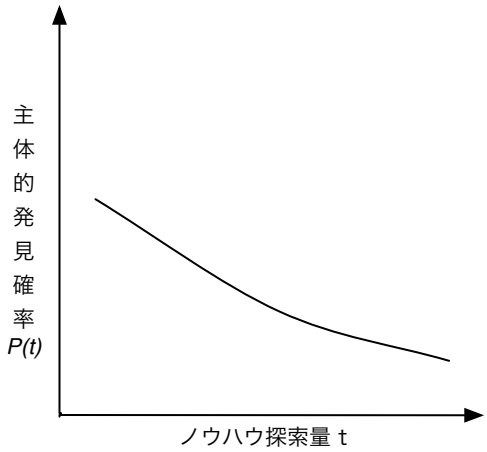


図 2.4: 探索量に対して逓減する主観的発見確率

ただし、探索活動においてユーザが求める情報に近い情報が見つかる場合、その主観的発見確率は高まる。近い情報が見つかるのであれば、ユーザが求めている情報が見つかるのではないか、という期待感が高まるためである。例えば、Web 検索などにおいては、求めているものに近い情報が多く見つかるために発見できるという期待は高まるのであるが、実際は見つからないという事も多い。このように、主観的確率は心理的な要素が大きく影響するため、必ずしも合理的な数値に近いものになるということはなく、またユーザの個人差も大きい。ここでの確率はあくまでも「主観的な」ものであることを意識しておく必要がある。

2.2.5 探索コスト予想値

外部ノウハウ探索の方法として Web で探索する、図書館で調べる、人に聞くなどのいくつかの方法がある。それぞれ情報源は異なっているが、外部の情報を取得するという意味では共通しており、抽象化したプロセスとして表すことができる。そのプロセスは図 2.5のように表す事が出来る。

外的情報にアクセスするためには、まず欲しい情報を手に入れられるような問い合わせを言語化しなくてはならない。次にその問い合わせを用いて、情報を探索し、情報へのアクセス手段を特定する必要がある。アクセス手段が特定できたら実際に情報へアクセスし入手する。そして手に入れた情報を解釈する。

外的ノウハウ探索にかかるコストは、図 2.5の各ステップに対して発生すると考えられる。そのコスト予想値の総和がユーザが知覚する「探索コスト予想値」である。探索コスト予想値は探索方略によって決まるため、ユーザがどの探索方略を選択するかはコストによって決まる事がおおい。

この探索コスト予想値も、ユーザが過去に探索方略を選択し実行する事で得られた経験に大きく影響する。つまり、2.2.4.2で述べた主観的確率と同じように、主観的なコスト予想値である。実際のコスト予想値と近いとは限らない。

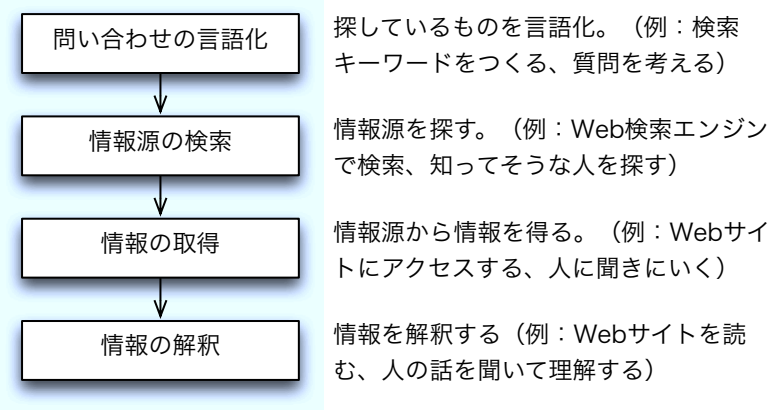


図 2.5: 外的情報の取得プロセス

2.2.6　外的情報探索方略の選択モデル

上記の議論を踏まえ、外的情報探索方略の選択モデルを構築する。これは図 2.2で紹介したグラフの外的情報探索部分を拡張したものとして表すことができる。(図 2.6参照)

2.2.2で説明したモデルに、複数の外的情報方略からどれを選ぶか、その意思決定をモデル化した部分を加えた。

2.2.7　本研究におけるノウハウ探索モデル

節 2.2.1、節 2.2.2で述べたモデルを統合し、本研究におけるノウハウ探索モデルを図 2.7のように構築した。

2.3　問題に対しての関連研究

1.3で取り上げた問題について、どのような研究分野からどのようなアプローチがなされているか紹介する。

2.3.1　外的ノウハウ探索未発生による機会損失問題へのアプローチ

外的ノウハウ探索未発生の問題に対しては 3 つの対応策がある。

1 つ目は効用増加期待値を適切にあげることである。これはさらに「追加で情報を手に入れたときに増加する効用の見込み」(節 2.2.4.1参照) を上げる方法と、「正しい情報が見つかる主観的確率」(節 2.2.2参照) を上げる方法がある。前者はユーザに「ノウハウが足りていないこと」を印象付けたり、外的ノウハウ探索が行われるたびに有効なノウハウを提供するといったことでユーザを「学習」させることで行える。後者は外的ノウハウ探索によってノウハウが必ず見つかると言う経験をつませ、ユーザからの信頼性を上げることで達成できる。

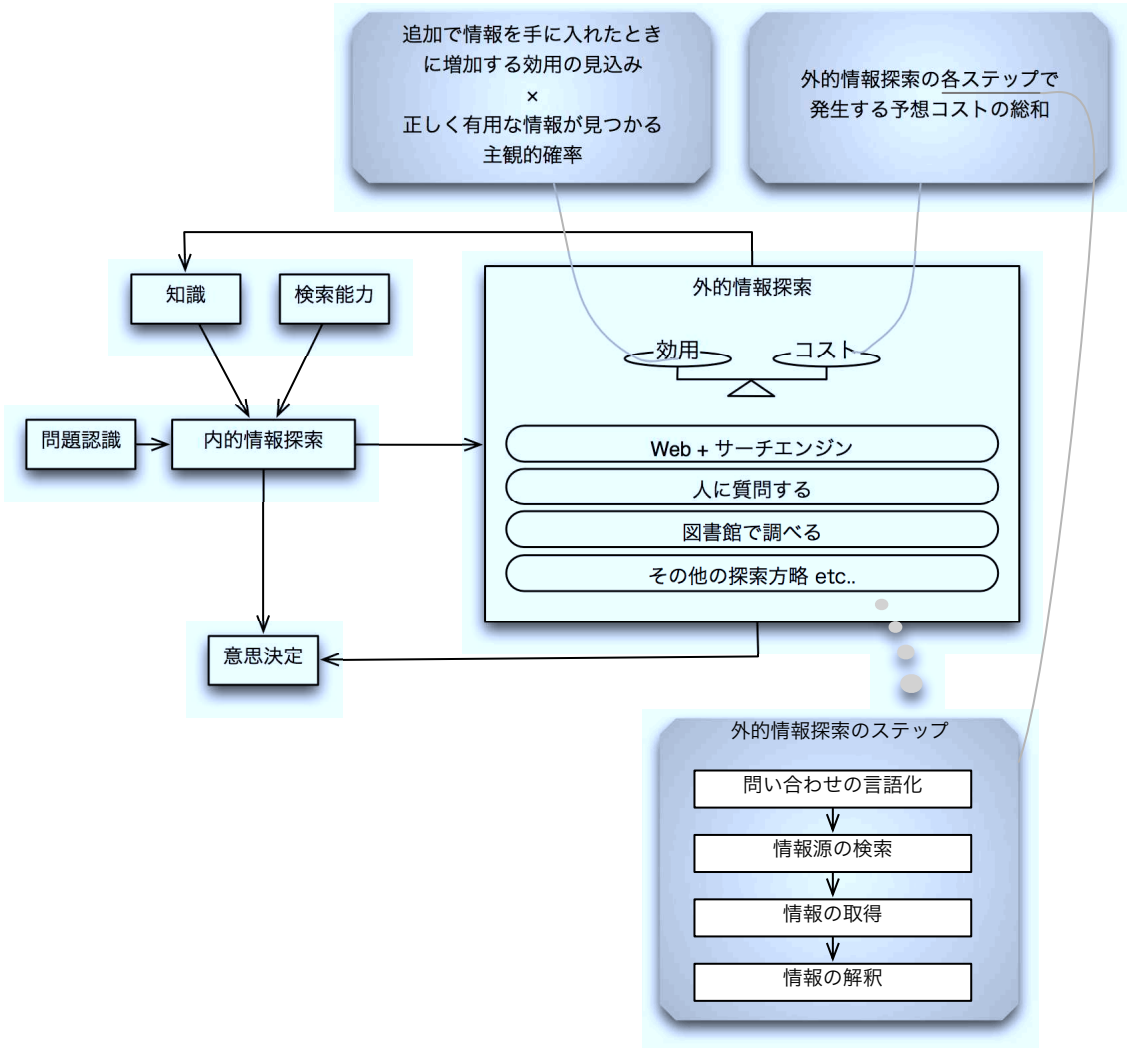


図 2.6: 外的ノウハウ探索方略の選択モデル

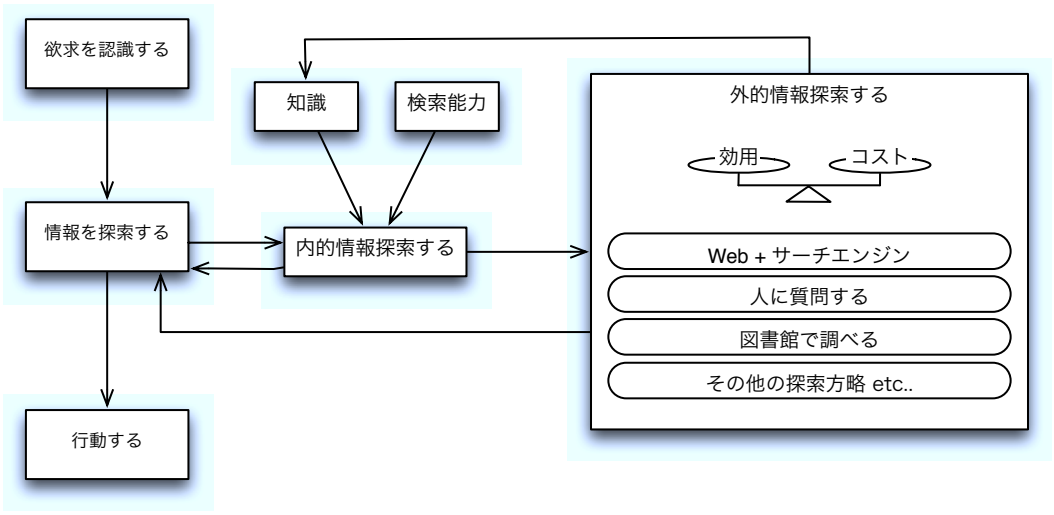


図 2.7: 本研究におけるユーザのノウハウ探索モデル

2つ目は探索コスト予想値を下げることである。ユーザの「探索コスト予想値」(2.2.5参照）を減らすためには、外的ノウハウ探索の各ステップ（図 2.5）にかける各コストを削減し、何度も利用してもらうことでユーザにコストが低いことを「学習」してもらう必要がある。

これら 2 つの対応策において、Web 検索エンジンは大きな成功を果たしている。膨大な Web を全て検索対象とすることで、求めているノウハウを発見できるとユーザに信じさせている。さらに探索にコストもキーワードの入力と数回のクリックで済むという手軽さを実現している。

Web 検索エンジンの台頭により、ユーザが外部情報探索を実施するコストが下がり、行動するときには検索するというのが新しい習慣となりつつある。電通は、旧来は内的情報探索のみによって行動が決定されていたが、Web 検索エンジンの普及に伴い消費者の行動モデルに「サーチ」というステップが組み込まれたと指摘している。[7] 今まではユーザーは「Attention」、「Interest」、「Desire」、「Memory」、「Action」という行動プロセスを踏んで購買行動を起こしていると分析されていた。しかしサーチエンジンの登場により、「Attention」、「Interest」、「Search」、「Action」、「Share」という行動プロセスを踏むことで行われようになったと分析されるようになった。

このように「効用増加期待値を上げる」、「探索コスト予想値を下げる」といったアプローチにおいて Web 検索エンジンは成功を収め、ユーザの行動パターンに変化をもたらしたと分析される。

外的ノウハウ探索をユーザに起こさせる第 3 のアプローチは、ユーザの状況にあわせて情報を「プッシュ型」で提供するという方法である。従来の外的情報探索はユーザが求めたときしか実行されなかった。Web はユーザが求めたときのみサービスが提供される「プル型」であり、Web 技術の上に構築された Web 検索エンジンも「プル型」である。ユーザの状況と抱えているタスクに合わせてシステムが情報を提供することができれば、ユーザがノウハウに出会う機会を増やすことができる。このように「RFID などの技術を用い

て人間が意識せずとも、水面下でコンピュータが能動的にデータを収集・処理を行い、情報化する技術及びそれらの概念」のことをコンテクストアウェアネスと呼ぶ。[8]

しかし、「習慣化により外的ノウハウ探索が行われにくくなる」という傾向は人間の思考力という貴重な資源を節約し、行動を迅速にするためのメカニズムであるとも考えられる。すべての行動についてより良いノウハウを探索していると時間が足りず、ユーザが処理するべき情報が膨大になり混乱を招く。この「習慣化による内的ノウハウ探索への依存」の解決には、ユーザに新しい情報を表示しすぎて混乱させることがないようにする配慮が求められる。

2.3.2 未言語化ノウハウの探索問題へのアプローチ

未言語化ノウハウは、人々の脳の中に記憶として存在するため、コンピュータで直接検索できない。

未言語化ノウハウを探索する方法として「質問掲示板」、さらにそれを進化させたものとして「Q&A サイト」が上げられる。「質問掲示板」は BBS の仕組みを用い、質問を掲載することで多くの人に見てもらい、知識のあるユーザに回答してもらう、という仕組みである。ユーザが質問に答えてくれそうな回答者を探すのではなく、回答者が質問に答えるという形式である。Q&A サイトは「質問掲示板」の仕組みをさらに進化させ、質問の検索機能を強化したり、ポイントシステムを導入することで回答者のインセンティブまで考慮に入れた仕組みである。これらの仕組みは一度回答された質問を文書化として保存することができるので、次からはその未言語化ノウハウも検索の対象になる。

掲示板や Q&A サイトは回答者ユーザに質問に気付いてもらわなければならない。また、ノウハウを言語化してもらう、という負担を強いているため、回答者が必ず回答するというわけではない。そのため、質問してもノウハウ回答者が見つからないことも多い。

そこでもうひとつのアプローチは人間の行動履歴を解析することで、その人間の蓄えているノウハウを導き出すという方法である。そのように、人間の行動履歴から有用な情報を導き出すだそうという試みを「ライフログ」という。[11]

2.3.3 欲求認識における背景知識不足問題

この問題を解決するためには、ユーザの抱えている問題を的確に把握するための仕組みが必要である。ユーザが認識している欲求が、その欲求の裏に潜んでいる真の欲求と照らし合わせて妥当なものであるか理解する必要がある。

ユーザから検索コマンドの入力をうけてから検索を開始するシステムでは、このような仕組みを構築するのは非常に難しい。[1] このような問題を解決する最も手っ取り早い解法は、「人に聞く」ということである。人に自らの大きな目的（背景に潜む欲求）から説明し、具体的にどのようなことを問題にすれば良いか聞くのが最も簡単である。

この問題では、自らのもつ欲求の裏にある、真の欲求をシステムに認識させ、さらに「人に聞く」ことのコストを下げるシステムがあれば、適切なノウハウの適用が可能になる。

[1]Web 検索の Google には、過去の検索データから統計的に推測して、ユーザが入力したキーワードと共に検索される事が多いキーワードを表示する「サジェスション機能」がある。

第3章　方法

1.3で指摘した問題を解決するためのシステム「Shumu」の基本的なアプローチを説明する。

以下の節で Shumu の特徴を説明する。

3.1　ToDo リスト管理ソフトとしての実装

Shumu はユーザの行動計画を支援する ToDo リスト管理ソフトとして実装する。ToDo 管理ソフトは「やること」を登録し、その処理状況を管理するソフトウェアである。日常的な行動管理や行動計画のために一般的に用いられている。

Shumu が ToDo リスト管理ソフトとして実装されるメリットは 2 つある。

1 つ目は ToDo リスト管理ソフトとして実装することで、ユーザの行動計画のタイミングにちょうど情報を表示することができることである。ToDo リストに新たな ToDo を記録するとき、人はその行動をどのように実行するかを模索している段階であると考えられる。ノウハウ探索が発生する行動計画のソフトウェアとして実装することで、行動計画のタイミングでリアルタイムにノウハウ探索を行い、「プッシュ型」でノウハウの表示提供を行える。

また、2 つ目の理由としてノウハウ探索支援システムとして新たなソフトの概念を把握せずに使いこなすことができるためである。システムを使うユーザが新しい探索システムを使うのを 0 から覚えるのは大変だが、既存の ToDo 管理ソフトの概念を拡張することで、比較的抵抗感なくシステムの利用を開始できると考えられるためである。

3.2　ToDo データをタスク認識と行動ログとして用いる

ToDo のうち未完了のものはこれから「やること」、まさにユーザが現在抱えているタスクを示している。ユーザに適したタイミングで適した情報を提供するコンテクストアウェア・システムでは、そのユーザが置かれている状況に加え、抱えているタスクを検知することが必要である。コンテクストアウェアの研究では、ユーザの置かれている状況を検出するためにさまざまな技術が使われている。しかし環境情報からユーザの抱えているタスクを汎用的に高い精度で検知する手法は確立されておらず、ユーザのタスクを認識することは大きな課題である。もし「未完了 ToDo」がユーザの抱えているタスクを示していると考えられるならば、ユーザにタスクを新たな負担なく入力させることができ、コンテクストアウェア技術を補完することができる。ToDo リスト管理をユーザがすでに行う習慣をもっているならば、ユーザにシステムを操作する負担を新たにかけることなく、ユーザのタスクを検出できる。

さらに、ユーザが処理し終わった ToDo は「やったこと」を意味しており、それは行動ログ情報として用いることができる。行動ログにはユーザがタスクを解決したときの手順が記録されており、ノウハウと呼べる情報が詰まっていると考えられる。ユーザの行動ログを元に有用な情報を生み出し、活用する研究は「ライフログ」という研究分野として位置づけられる。ライフログでは、集められたユーザの行動記録について、「意味づけ」する、つまり、「どのような意思を持ってその行動が行われたのか」を検出するのかが 1 つのテーマになっている。Shumu ではユーザの入力した ToDo にはどのような意図で行動を行うのかという情報も含まれており、ライフログの研究分野を補完することができると考えられる。

つまり、ToDo リストのデータのうち未完了のものは「ユーザが抱えているタスク」、完了したものは「ユーザの行動ログ」として認識する事ができる。

3.3 他者の行動ログを表示する

ユーザが行動を起こすときに、その行動を他者がどのように計画し実行したか、その行動ログを表示する。Shumu では、ユーザが ToDo を入力し、その ToDo をどのように実行しようかと考えているタイミングで、他者がその ToDo を過去にどう処理したかを表示する。Shumu のこの他者の行動ログ（ToDo データ）を表示する機能を Shumu では「行動事例表示機能」と呼ぶ。

この「他者の行動ログを表示する」というアプローチには 2 つの目的がある。「他者の『模倣』の促進」と、「ノウハウをもつユーザの検索」の実現を目指す。

1 つ目の目的である「他者の『模倣』の促進」とは、ユーザに他者の ToDo を表示する事で行動計画の参考にしてもらうことを目指している。人間は他者の行動を見て模倣する事が出来る。模倣には他者の行動から、その行動の意味や目的を察し、自分の行動にどのように当てはめるべきかという高度な処理が求められる。知識処理の分野において、論理演算で人間の抱えるタスクを推測したり、その行動の意味を推察するという研究が行われているが成功を収めているとはいい難い。[1]

本研究では他者が理解する事を前提に作られていない ToDo リストからであっても、ユーザはノウハウを抽出し、自らの行動に当てはめることも「模倣」の一種であると考える。そして、以下の仮説を提案する。

仮説 1 (ユーザは他者の ToDo リストを見て模倣する事が出来る)**.** *Shumu* で表示された「他者の *ToDo*」事例を見る事で、ユーザはその *ToDo* のノウハウを抽出し、自らの問題に適用する事が出来る。

「他者の『模倣』の促進」は、ユーザがその問題についてある程度の知識を持っていることが前提である。ある程度の知識があれば、自らのおかれている状況に必要な情報のみを選択したり、足りない部分を補う事ができる。しかし、まったく背景知識のない問題については、他者の行動ログを見ても、どこをどう模倣すれば良いのかも分からないだろう。そのような場合でも、経験を持っている他者に聞く事が出来れば、問題に対して探してい

[1] 典型的な問題として、人間は問題に関連する知識を一瞬で選別する事ができるが、コンピュータには難しいということを指摘する「フレーム問題」がある。

るノウハウが妥当なものであるか、どういった知識が必要であるかなど、相談することができる。

そこで Shumu の行動事例表示機能は、ユーザが必要としているノウハウを誰が持っていて、誰に聞けば良いのかを明らかにすることを 2 つ目の目的として目指している。過去にある行動を起こしたユーザは、その行動についての経験をもっていると考える事が出来る。ユーザは経験を持つ他者を検索できれば、その他者に聞く事でノウハウを獲得する事が出来る。

元来、「人に聞く」というノウハウ探索方法は、誰に聞けば良いか探すのにコストがかかる。Web 検索に比べ、「正しい情報が見つかる主観的確率」が低いためにノウハウが見つかるかどうか分からない。また、ある人の脳内にノウハウが存在するかは質問する事でしか確認できず、知っていそうな人を探し聞いて回るしかなく、探索コストが高いと言える。そのため、「人に聞く」というノウハウ探索処方は Web 検索に比べて行いにくかった。しかし、自分がやりたいことについて、誰が経験を持つかをユーザに提示する事が出来れば、ユーザがその他者に聞くというノウハウ獲得のコストを下げる事が出来る。

3.4 ToDo を分解して入力できる UI を持たせる

Shumu では、タスクはいくつかのタスクに分解できるものであると考える。分解されるタスクを上位タスクと呼び、分解された上位タスクを構成するタスクを下位タスクと呼ぶ。上位タスクは下位タスクを実行する事で達成されるものである。このことを上位タスクは下位タスクと「is-achieved-by 関係」であるという。

ToDo データを分解しツリー構造として入力できるようにする事で、2 つのメリットがある。

1 つ目はユーザがタスクの構造を把握しながら計画を立てられるようになることである。人はタスクを遂行するために、タスクを実行できるサイズの小さなタスクに分解してから実行する事がある。プロジェクトマネジメントにおいては、プロジェクトの目標を具体的に実行可能なサイズになるまでタスクを分解する事を「WBS(Work Breakdown Structure」と呼ぶ。このようにタスクを分解する事で、人はタスク全体のタスクをもれなく洗い出す事ができ、仕事の正確性や効率性を上げる事が出来る。タスクの把握において、人間は分解・整理できたほうがよいと考えられている。

2 つ目のメリットは上位タスクと照らして妥当なタスクであるか検証できるようになることである。ユーザに十分な知識がない場合は、誤って上位タスクを達成するのに妥当でないタスクに分解していってしまう事がある。上位タスクと下位タスクの「is-achieved-by 関係」が妥当でない場合がある。このようなときに、Shumu では上位タスクの情報を用いて他者の分解例をしめすことで、下位タスクの設定が間違っている可能性をユーザに示すことができる。

Shumu では、この ToDo を分解する時が行動計画のステップであると考え、このタイミングで他者の行動ログを表示する。

3.5　社会的距離が近いユーザのノウハウを検索する

それぞれのユーザがノウハウを持っており、それぞれのユーザが他者のノウハウを探しているとする。そのときに、ユーザが求めているノウハウはどのように検索されるのが良いか考察する。

ユーザのおかれているタスク環境には、２つのものが含まれている。タスクの目的と、問題状況である。このとき、タスクの目的は一度明確に決めれば普遍的に共通する。しかし、現状や問題を解決するために使えるリソース、または問題解決の考慮するべき社会的慣習などが含まれる問題状況は、それぞれのユーザで大きく異なる。そのため、2 人のユーザのタスクがそれぞれ同じ目的をもつとして認識されたとしても、ユーザのおかれた問題状況が異なるために全く異なる内容になるということは頻繁に起こる。

そこで Shumu では 1 つの仮説をおく。

仮説 2 (社会的距離の近いユーザであれば抱えている問題状況も近い)**.** 社会的に距離が近いユーザは、お互いに社会的環境や慣習などが近く、タスクの問題状況なども近い傾向がある。そのため、ユーザの社会的距離が近ければ近いほど、お互いのノウハウが参考になる可能性が高い。

この仮説に基づき、Shumu で他者の行動ログを検索するときは、社会的距離の近いユーザの物から探す。

タスクには２つのタイプがある。「形式的なタスク」と「非形式的なタスク」である。

「形式的なタスク」は問題状況がよく定式化されているものである。例えば「数学の問題『1+1』を解く」といった問題などである。これらのタスクは問題状況が多くの人にとって共通する。数学の問題は高度に抽象化され、問題状況が万人にとって共通するように作られた、「形式的なタスク」の最たるものである。

「非形式的なタスク」は問題状況に含まれる情報が多く、問題状況を切り離して考えることができない。例えば「おいしいレストランを探す」といった問題である。このようなタスクは問題状況がそれぞれの人で異なる。おいしいの定義は人それぞれであり、さらに都合がいい地理的条件や使う事ができる交通手段なども人によって異なる。そういう問題に関しては普遍的な正解は存在せず、問題状況に応じた行動計画が求められる。このような非形式的なタスクのノウハウは、そのユーザの抱える問題状況にあったものが求められる。

Shumu では、非形式的なタスクについて問題状況を考慮しなくてはならないという問題を、社会的距離が近いユーザは抱えている問題状況も近いと仮定することで対応する。

第4章　設計

本研究ではノウハウの探索を支援するシステム「Shumu」を実装した。本章ではその設計について述べる。

4.1　Shumu の構成

Shumu は主に 3 つのモジュールから構成される。

- ユーザインターフェイス
- データ
- マッチング

Shumu の基本的なモジュール設計を図 4.1に表す。図の上部の矢印はユーザとのやり取りを意味し、図右下部の矢印は他ユーザとの通信のやり取りを表している。

ユーザはユーザインターフェースモジュールに対して ToDo を入力する。その ToDo データはデータモジュールに保存される。ユーザインターフェイスモジュールはユーザが ToDo を分解中である事を検知すると、その分解中の ToDo に近い ToDo がないかマッチングモジュールに問い合わせる。マッチングモジュールは自ユーザの過去の ToDo 事例をデータモジュールから検索すると同時に、自分と社会的距離の近い他ユーザの ToDo 事例の中からも検索するべく、他ユーザのマッチングモジュールに問い合わせクエリーを発行する。マッチングモジュールはそのようにして集めた ToDo 事例をユーザインターフェイスモジュールに返し、ユーザーインターフェイスモジュールはユーザに行動事例を表示する。

また、他ユーザからの問い合わせクエリーを受けた場合、自らの過去の ToDo 事例を検索し適切な ToDo 事例を返信する。この他ユーザからの問い合わせに対しての返答は自動的に行われ、問い合わせを受けたユーザは Shumu に対して特段の操作を行う必要がない。

以下ではそれぞれのモジュールについての設計を説明する。

4.2　ユーザインターフェイス

ユーザーインターフェースモジュールはユーザからの入力を受け付け、情報を表示するモジュールである。そのモジュールはさらに ToDo の入力と表示、分解などの基本的な操作を受け付ける「ToDo 分解サブモジュール」と、ユーザが ToDo 分解中に他者の行動事例を検索し表示する「行動事例表示サブモジュール」に分けられる。

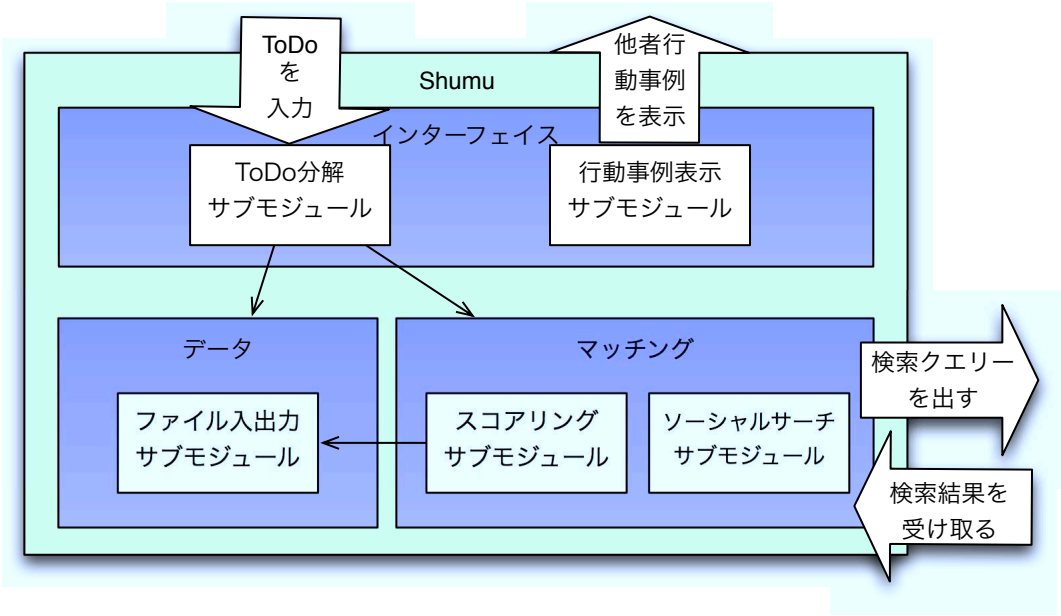


図 4.1: Shumu モジュール図

4.2.1 ToDo 分解サブモジュール

ToDo 分解サブモジュールはユーザが ToDo の入力・分解作業を行うモジュールである。ToDo リストに近いインターフェイスを持ち、ユーザにとって直感的で分かりやすいものを目指している。ToDo 分解サブモジュールはユーザが直接的に扱うモジュールであり、ユーザからみたユースケースは図 4.2に表せる。

Shumu の ToDo リストは図 4.3のような形をしている。

ユーザがリストに ToDo を追加したいときにはリスト下部の「＋」と示されたボタンをクリックする事で、追加する事が出来る。(図 4.3参照)

ユーザは ToDo をダブルクリックする事で、編集モードにする事が出来る。編集モードでは ToDo の内容がテキストボックスに入力されており、それを書き換える事で ToDo の内容を書き換える事が出来る。ToDo を削除する場合は編集モードでの左下部に表示される「削除」ボタンを押す事で削除できる。ToDo の内容を書き換えた場合、編集モード右下部の「更新」ボタンを押す事で更新できる。(図 4.4参照)

ユーザは各 ToDo の右に表示されている長方形のボタンをクリックする事で、その ToDo を遂行するために必要な下位 ToDo を入力する ToDo リストを表示させる事が出来る。この長方形のボタンを「リスト開閉ボタン」と呼ぶ。ToDo リストは上位 ToDo のリスト開閉ボタンを押す事で、開閉する事が出来る。このとき内容があったリストを閉じても、内容は失われず再度開く事で表示する事が出来る。下位 ToDo が開いた ToDo リストを持っていたとしても、下位 ToDo の末端まで波及して閉じられる。このリスト開閉ボタンはユーザが表示したいリストのみを表示し、一覧性を高めるためのものである。上位 ToDo とその ToDo を達成するための下位 ToDo のリストは赤い線で結ばれ、どのような繋がりがあるか視覚的に把握できる。ToDo リストの位置はそれぞれが重ならないように配置され、

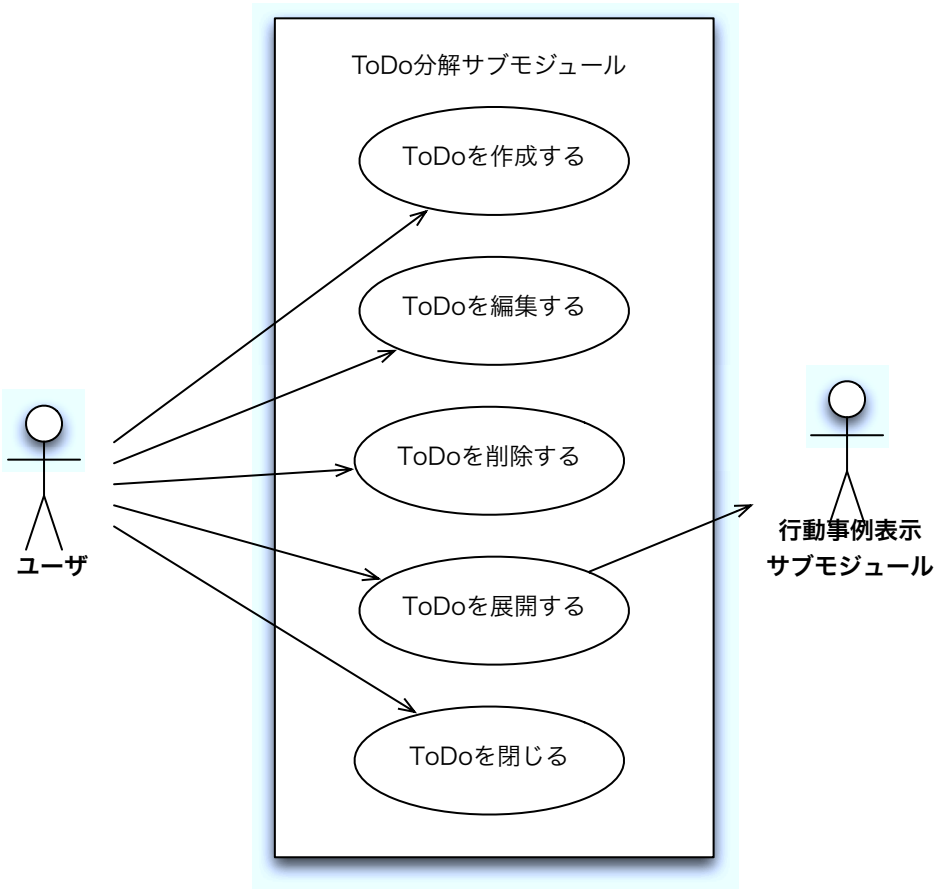


図 4.2: ToDo 分解サブモジュールのユースケース

図 4.3: ToDo リスト

図 4.4: 編集モード

開いている ToDo リストや ToDo リストのサイズに変更があった場合は動的に再配置される。(図 4.5参照)

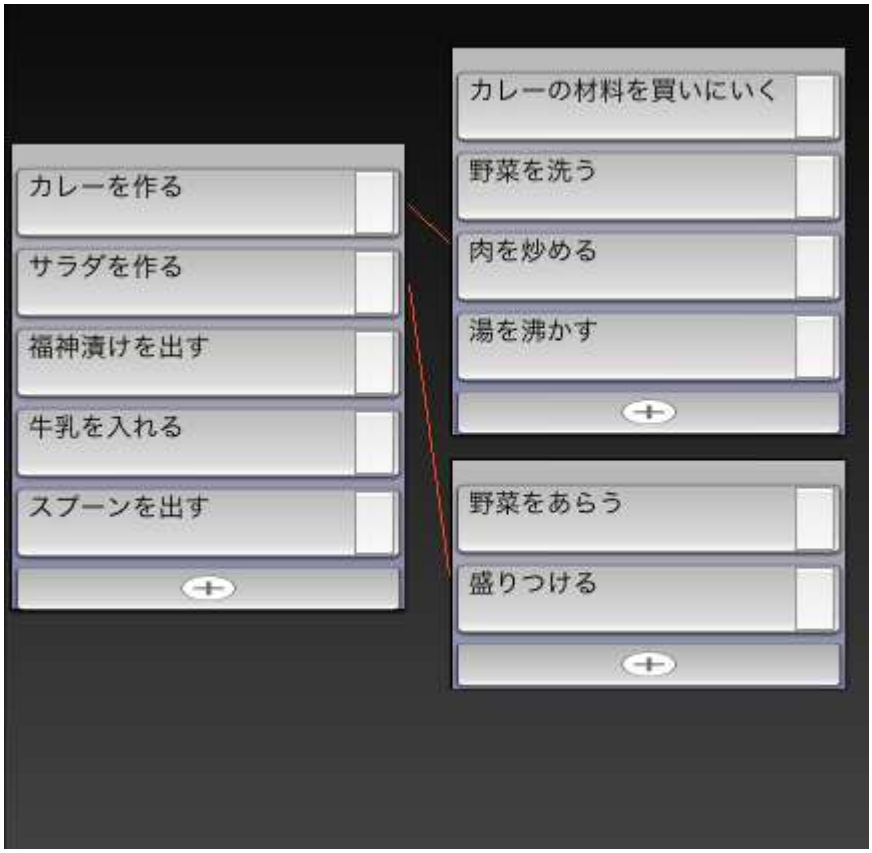


図 4.5: ToDo リストを開いた様子

このように Shumu では ToDo をツリー構造として扱い、そのツリー構造は大きくなる可能性がある。そのためツリー構造が巨大であっても把握できる、一覧性の高いインターフェイスを構築する必要がある。Shumu では、大きなキャンバスを作りその上に ToDo が表示される形式にし、ユーザはそのキャンバスをスクロールして全面を移動する事ができるようにした。

4.2.2 行動事例表示サブモジュール

行動事例表示サブモジュールは検索結果をユーザに表示するモジュールである。

ユーザが ToDo のリスト開閉ボタンを押してリストを展開した時、ユーザはその ToDo を分解しようとしていると考える事が出来る。展開されたタイミングで ToDo 分解サブモジュールは自分の過去の行動ログや他者の行動ログの中から参考になりそうな行動事例を検索するために、マッチングモジュールに対して検索を依頼する。マッチングモジュールは見つかった行動事例を、「行動事例表示サブモジュール」に送る。

行動事例表示サブモジュールは送られてきた行動事例を、「行動事例候補一覧ボックス」に表示する。(図 4.6) 行動事例候補一覧ボックスの上部には「分解中の ToDo」が表示さ

図 4.6: 行動事例候補一覧ボックス

れている。その下のリストにはマッチングモジュールより送られてきた行動事例検索結果がスコア順にソートされて表示されている。

リストの中の項目をダブルクリックすると、図 4.7のような「行動事例ビューウィンドウ」が開き、行動事例が表示される。ユーザは行動事例ビューウィンドウに表示された過

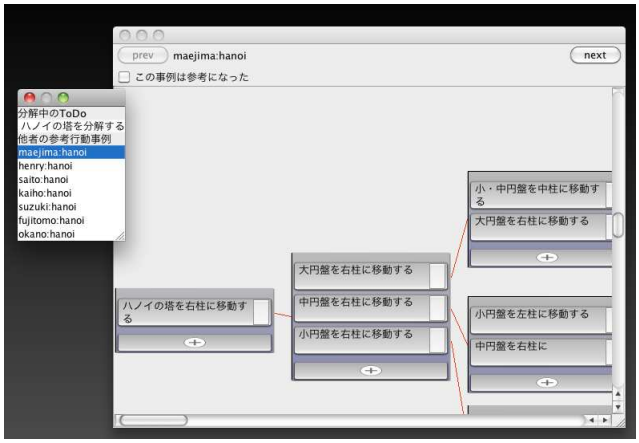


図 4.7: 行動事例ビューウィンドウ

去の自分の行動ログ、他者の行動ログを参照しながら自らの抱えている ToDo を分解する。ToDo 分解画面と同じ構造をしているが、行動事例ビューウィンドウに表示された ToDo

は追加や削除などの編集をする事はできない。

4.3 データ

Shumu のデータモジュールはユーザが入力した ToDo をファイルに保存し、そのファイルにアクセスする機能を持つ。ユーザインターフェイスモジュールやマッチングモジュールから指示を受けた場合、その指示に基づいて過去の ToDo データを返す。

Shumu の ToDo は最低限下記のフィールドをメンバとして持つ。

- ToDo テキスト
- 下位 ToDo のリスト

下位 ToDo のリストの中には複数の ToDo を持つ事ができる。実用的なシステムになるためには、ToDo に締め切り日や進捗状況のデータ構造を付加する必要があるが、本システムでは実験段階のため、そこまでの情報を持たせていない。しかし実装で述べるように、Shumu では XML 形式で ToDo を扱っているため、今後拡張する事が出来る。

ToDo は 1 つのプロジェクトごとにファイルを分ける。各プロジェクトの最も上位には「ルート ToDo」という ToDo テキストが空の ToDo が設定されている。

ファイルのデータ構造の例を図 4.8に示す。

4.4 マッチング

ユーザインターフェイスモジュールよりユーザが分解中の ToDo を受け取り、参考になりそうな行動事例を探し出すのがマッチングモジュールに求められる機能である。方法論にて説明したように（節 3.5参照）、Shumu では過去の自分の行動ログの中からのみならず、社会的距離の近いユーザである「フレンド」の保有する行動ログのなかからも検索しようとする。

「フレンド」は、Shumu がユーザと直接の交友があると判断する他のユーザのことで、ソーシャルグラフ上でユーザから 1hop でつながれるユーザである。Shumu はこのフレンドを直接的には定義せず、既存のソーシャルメディアの上に構築されたソーシャルグラフを用いる事を想定している。つまり、フレンドの厳密な定義は実装に依存する。現状の Shumu では Jabber インスタントメッセンジャー上で「コンタクトリスト」に入っているユーザをフレンドとしている。

Shumu のマッチングモジュールは 2 つのサブモジュールから構成される。1 つ目は ToDo のテキストを直接解釈して類似度を判断する「スコアリングサブモジュール」、2 つ目はフレンドとの問い合わせや返答をおこなう「ソーシャルサーチサブモジュール」である。

Shumu が行動ログを検索する時、自らの過去の ToDo リストをスコアリングすると同時に、ソーシャルサーチも行っている。スコアリングサブモジュールとソーシャルサーチサブモジュールに同時にクエリーを送る。

```
<?xml version="1.0" encoding="UTF-8"?>
<task>
    <tasktext></tasktext>
    <child-tasks>
        <task>
            <tasktext>カレーを作る</tasktext>
            <child-tasks>
                <task>
                    <tasktext>カレーの材料を買いにいく</tasktext>
                </task>
                <task>
                    <tasktext>野菜を洗う</tasktext>
                </task>
                <task>
                    <tasktext>肉を炒める</tasktext>
                </task>
                <task>
                    <tasktext>湯を沸かす</tasktext>
                </task>
            </child-tasks>
        </task>
    </child-tasks>
</task>
```

図 4.8: ToDo の XML データ構造

4.4.1 スコアリングサブモジュール

スコアリングサブモジュールはユーザが分解中の ToDo と任意の ToDo の近似度を計測するためのモジュールである。具体的には与えられた 2 つのモジュールの近似度を計測して返す機能を持つ。Shumu では、それぞれの ToDo は単純なテキストとして記録されており、そのテキストを比較する事で近似度を計測する。比較の方法は、テキストの中に含まれる単語の一致数で計る。

しかし単純な単語の一致数では情報量が少なく、精度の高い比較は難しい。そこで Shumu では、それぞれの ToDo のもつツリー構造の情報も用い、精度を高める工夫をしている。

ToDo のツリー構造に注目すると、上位 ToDo を達成するために下位 ToDo を分解していることが分かる。つまり、下位 ToDo を分解している時に、上位 ToDo はユーザのおかれている環境を構成していると解釈する事が出来る。そこで、Shumu では上位 ToDo から下位 ToDo までの全てのテキストを使う事で、比較に用いる情報量を増やす。上位 ToDo から下位 ToDo までのテキストの道筋のことを、ディレクトリ構造におけるルートディレクトリからカレントディレクトリまでの道筋の事を意味する単語になぞらえて、「フルパス」と呼ぶ事にする。

ToDo のフルパス同士で比較することを例を用いて説明する。

例えば、葬式に行くときのネクタイを購入するという ToDo を考えてみる。葬式では黒いネクタイが必要であり、結婚式では白いネクタイを購入しなければならない。このとき、葬式に行こうとしているユーザは他の「葬式に行こうとしてネクタイを購入したときの行動ログ」を参考にして黒いネクタイを購入するべきであり、「結婚式に行こうとしてネクタイを購入したユーザの行動ログ」を参考にして白いネクタイを購入してはならない。

もし ToDo のフルパスで比較すると、葬式に行こうとしてネクタイを購入する ToDo 同士は結婚式の ToDo よりも高い「単語の一致数」を示す事が分かる。(図 4.9参照)

なお、同じ単語が複数回ヒットしても、1 つの単語としてカウントする。また、助詞などの ToDo の内容を説明していない単語についてはこの比較においては用いない。

この比較アルゴリズムは非常に単純で、フルパスを用いたとしてもあまり高い精度は期待できない。しかし、Shumu では最終的に参考にする ToDo を判断するのはユーザであるため、多少の誤りがあったとしてもユーザにとっては有用であると考えられる。

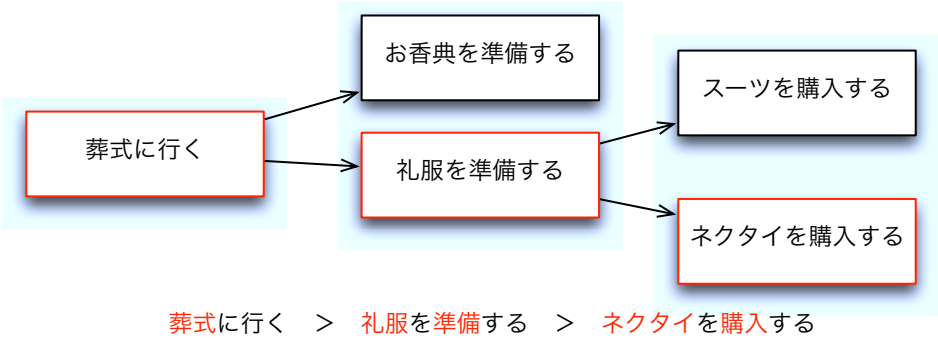
4.4.2 ソーシャルサーチサブモジュール

ユーザの分解している ToDo に近い行動ログを、社会的距離の近いユーザの中から探し出す機能を実現するのがソーシャルサーチサブモジュールの機能である。ソーシャルサーチサブモジュールの振るまいは、ユーザが行動ログを求めて問い合わせを行うとき（クライアントになるとき）と、行動ログの問い合わせに対して返答するとき（サーバーになるとき）の 2 つのパターンがある。

クライアント側として振る舞ったときは、次のステップを踏んで行われる。

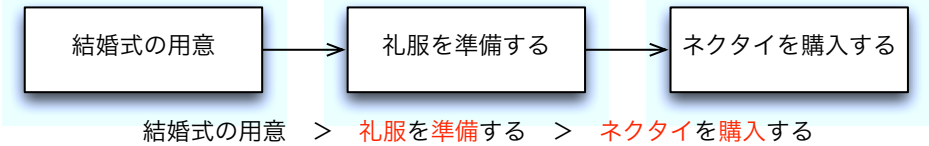
1. ユーザはオンラインの「フレンド」に自分が分解中の ToDo を送り、照会する。
2. 「フレンド」は送られてきた ToDo と自らの行動ログを比較し、近似度が一定以上のものを返答する。

【比較の基準ToDo】葬式のためのネクタイを購入



葬式に行く　＞　礼服を準備する　＞　ネクタイを購入する

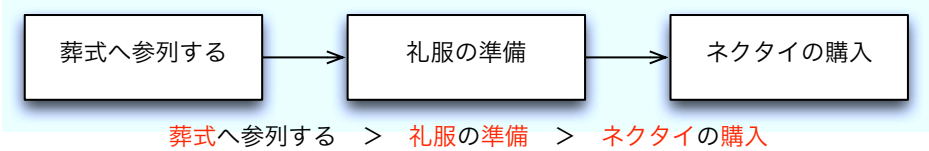
【参考にしては行けないToDo】結婚式のためのネクタイを購入



結婚式の用意　＞　礼服を準備する　＞　ネクタイを購入する

単語の一致数　＝　4

【参考にするべきToDo】葬式のためのネクタイを購入



葬式へ参列する　＞　礼服の準備　＞　ネクタイの購入

単語の一致数　＝　5

図 4.9: ToDo の比較アルゴリズム

3. ユーザは送り返された行動事例をスコア順にソートし、ユーザに表示する。

そのやりとりにおけるプロトコルは下記のように表す事ができる。

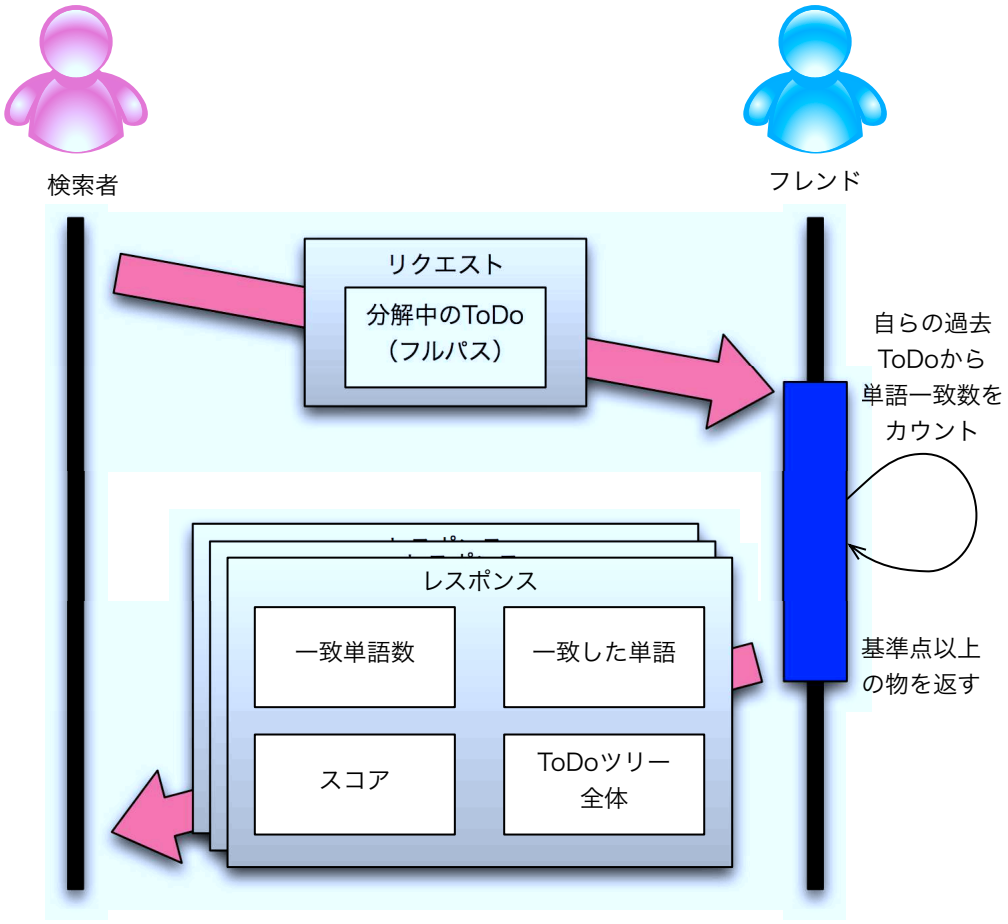


図 4.10: ソーシャルサーチのプロトコル

クライアント側はまず自らの「フレンド」の中からオンラインのユーザを選ぶ。さらにそのユーザが Shumu を使っているかどうかを判断し、使っているようであれば検索クエリーをおくる。検索クエリーは「現在分解中の ToDo」で構成されている。

サーバ側は、自ユーザに操作を求める事もなく、自動的に自らの行動ログを検索して結果を返答する。送られてきたクエリーに入っている ToDo と自らのもつ行動ログをスコアリングサブモジュールに送り、基準以上の単語の一致数を示した行動ログを一致した単語数、一致した単語、スコア、該当行動ログの ToDo ツリー全体を送り返す。（なお、現時点では「一致した単語数」と「スコア」は同義であるが、今後の拡張のために別の指標として扱っている。）

クライアントは検索クエリーを送ってから一定時間、サーバから返答が送られてくるのを待っている。返答があった場合、サジェスションボックスに追加していく。このとき、「スコア」の数値で降順にソートしながら追加していき、ユーザにとってより有用でありそうな ToDo を上部に表示させる。

第5章　実装

前章で説明した設計に基づき、本研究では ToDo リストを利用したノウハウ探索支援システム「Shumu」を実装した。Shumu はインスタントメッセージングプラットフォーム「wija」のプラグインとして実装した。ユーザは wija のプラグインとして起動し、日常的な ToDo 管理を支援するソフトとして構築した。Shumu を用いると、ユーザは日常的な ToDo 管理を行うときに自動で関連する他者の ToDo が表示され、参考にする、もしくはその他者に聞く事ができる。

5.1　実装環境

Shumu の実装環境を示す。

表 5.1: Shumu 実装環境

言語	Java
対象 JRE	1.4.2 以上
開発 javac バージョン	1.6.0-17
開発機	MacBook CPU: Core2 Duo 2.26GHz メモリ: 2GB

ユーザインターフェイスは Swing を用いて構築した。本研究では検証に必要な機能をもったユーザインターフェイス部分を構築した。

5.2　wija の提供する API の利用

Shumu の基盤として選択した wija は、XMPP に基づきメッセージングやチャットを行うインスタントメッセージングプラットフォームである。通信機能の実装、およびファイルの読み込み・保存に関わる XML パーサについては wija に備わっている API を用いた。wija では XML 形式で表されたデータをユーザ間で自由かつセキュアに行う事が出来るため、プラグインとして開発する事で通信機能を用いたアプリケーションの実装をより容易に行う事ができる。さらに XML の扱いを簡単にする API も用意されており、Shumu のデータ構造を保存する機能はそれを用いた。

図 4.1で説明した設計を、wija の API を用いて構築する事を考慮して作り直すと図 5.2のように表す事が出来る。

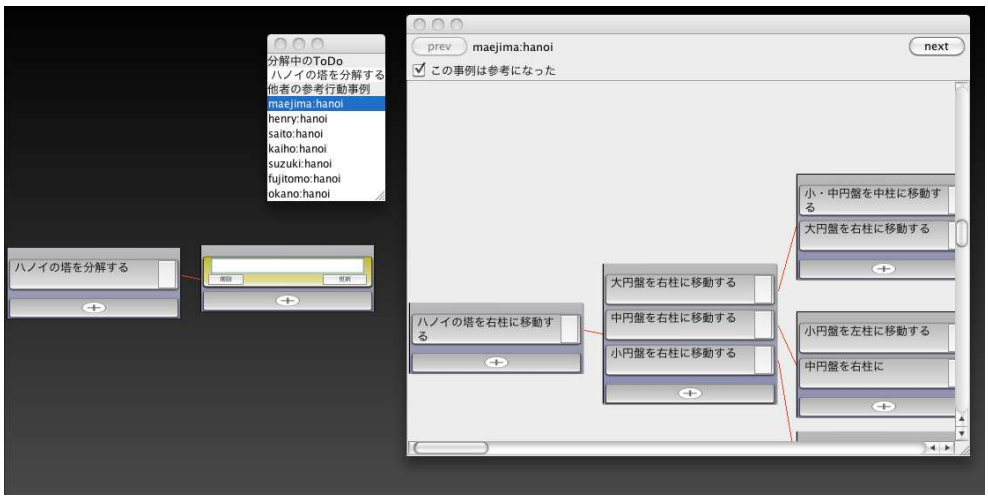


図 5.1: 実装された Shumu のスクリーンショット

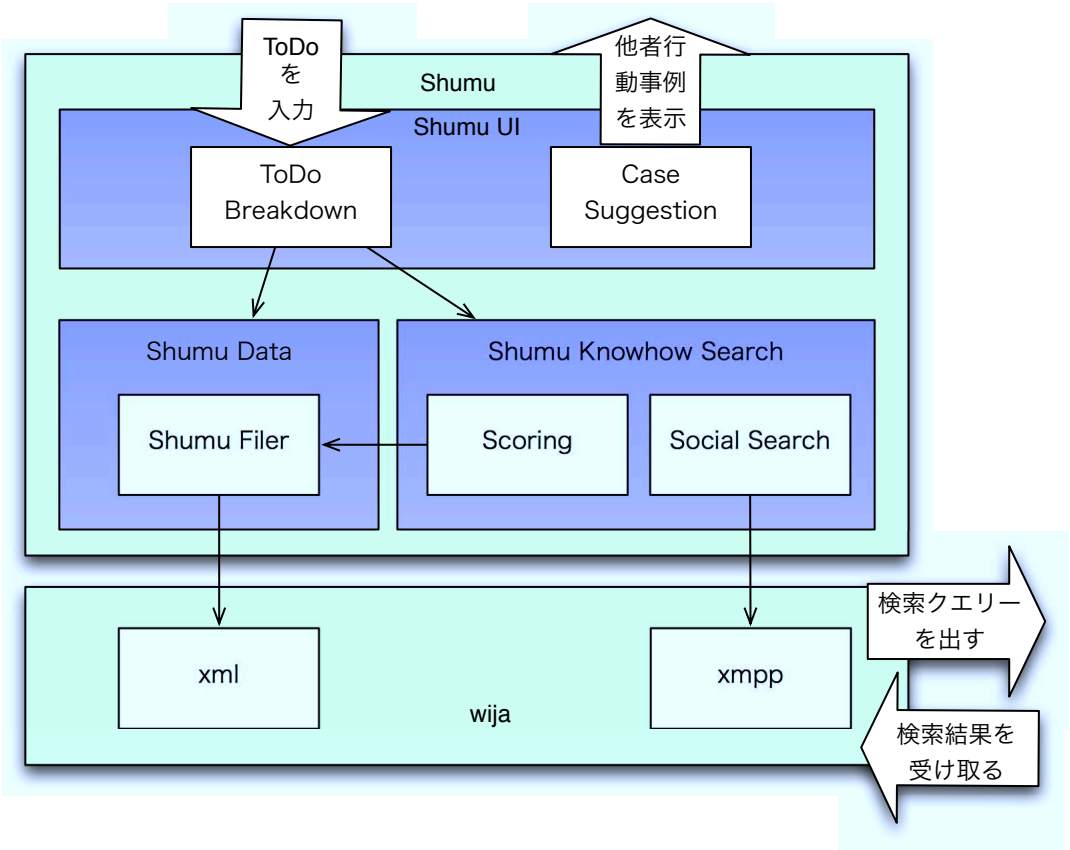


図 5.2: 実装におけるモジュール図

5.3 wija プラグインとして実装するメリット

wija のプラグインとして実装する事で、実装作業を簡略化するのみならず、他の wija プラグインとの連携を行う事ができる。本 Shumu ではノウハウの探索を行う事を支援するが、その取得については完全にサポートしきれていない。Web 検索エンジンであれば検索したデータに Web を用いてシームレスにアクセスする事ができるが、Shumu では直接「聞く」という手段を実現する手段を持たない。しかし、wija のインスタントメッセージング機能を用いる事で「メッセンジャーで聞く」というコミュニケーション手段を提供できる。

また、wija のプラグインの 1 つである電子通貨プラットフォーム「i-WAT」を用いる事で、ユーザ間のノウハウの決済を実現できる。Q&A システムでは、ユーザの回答を促すためにポイントシステムを併用するところがある。回答者がノウハウを言語化するには労力がかかり、インセンティブをどのように提供するかを考慮する必要がある。Shumu にはそのようなインセンティブを考慮した機能は存在しないが、i-WAT がその役割を果たす。

第6章　評価

Shumu を実際に利用してもらい、その方法論の評価を行った。

6.1　実験概要

実験は Shumu の「行動事例表示機能」（節 3.3参照）を使ったときと使わなかったときの２度に分けて行った。被験者たちに実際に 2 つの問題を与え、その問題を解く過程を Shumu の上で計画してもらった。

問題は２つ用意した。問題はそれぞれ、「形式的な問題」と「非形式的な問題」である。数学モデルや論理モデルなどに定式化できる問題は「形式的な問題」である。日常生活で処理する ToDo のほとんどは定式化できない「非形式的な問題」である。実験ではその二つをタイプの問題を出題した。

出題した問題は下記の二つである。

1. ハノイの塔を解く
2. 「ネットワークが繋がらない」と先生が言ってきたときの対応

「ハノイの塔」は図 6.1のように３本の柱に刺さった円盤を、左の柱から右の柱に全て移動するというパズルである。移動は一度に一枚だけでき、円盤の上にはそれより小さな円盤しか乗せられないという制約がある。本研究では円盤が３枚の場合の問題について解いてもらった。ハノイの塔は厳密にモデル化することが可能であり、数学的に解決法が示されている「形式的な問題」である。

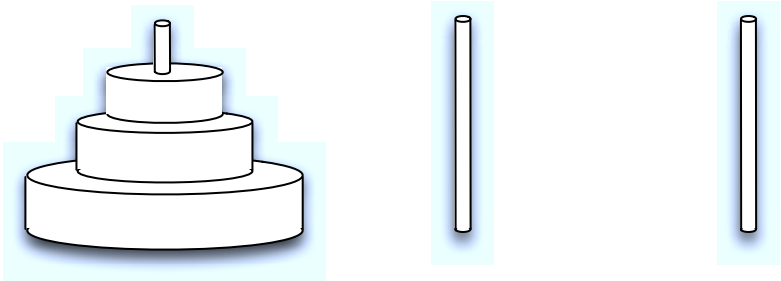


図 6.1: ハノイの塔

形式的な問題はユーザのおかれている環境に関わらず問題の理解がひとつである。そして問題にはひとつにさだまる答えがあるため、正解率を計ることができる。そのため、「ハ

ノイの塔を解く」という課題では、1 度目と 2 度目の正解率を測ることで、仮説 1 の示すとおり行動事例表示機能により ToDo リストの模倣が起こり、全体的な正解率が向上するという予想を検証する。

「ネットワークが繋がらないと先生が言ってきたときの対応」は、より現実に近い「非形式的な問題」である。このタスクでは被験者に「教師がネットワークが繋がらないと相談してきたとき、どのように行動するか」の対応を書いてもらった。このような非形式的な問題については正解をひとつに決めることはできない。それは、3.5において説明したように、問題の理解方法や、慣習による行動制限などの問題状況がユーザによって異なるためである。この 2 つ目の「非形式的な問題」では仮説 2 が Shumu の上で成り立つかどうか、検証を行った。被験者が 2 度目の「事例表示機能」を利用したときに、社会的距離が近いユーザ（フレンド）の ToDo 分解例を遠いユーザのものより参考になると評価するかどうかを検証する。

被験者は慶應義塾大学環境情報学部の村井研究室の 3 名と、同大学三田キャンパスのコンピュータ利用相談員 3 名の合計 6 名を選んだ。この 2 つの集団は非形式的な問題に対して、それぞれ異なった行動をとると思われる。村井研究室はネットワーク技術に関しての研究を行っており、彼らにとって「先生」はネットワークに対しての知識を持つ人間である。村井研の「先生」が研究室のメンバーに「ネットワークがつながらない」と言ってきた場合、ネットワークに問題がある場合がほとんどであり、ネットワークの状況を確かめ、問題の修復を行おうとするだろう。それに対して三田キャンパスは文系学部が集まる学部であり、コンピュータ利用相談員に質問する「先生」はネットワーク技術に対して知識を持たない。そのような「先生」が相談に来た場合、コンピュータ利用相談員が最初に疑うのはコンピュータの設定である。このように 2 つの集団で問題の背景が異なり、異なった ToDo 分解の仕方を行うのではないか、と考えた。

Shumu を操作するコンピュータは開発にも使った MacBook を貸与し、それを操作してもらった。

6.2 実験手順

実験は下記の手順で行った。

1. 被験者に Shumu の概念を説明する
2. Shumu を実際に操作してもらい、操作に慣れてもらう
3. 実験について説明する
4. 問題 1 を解いてもらう
5. 問題 2 を解いてもらう
6. 問題を解いてみての感想を聞く

実験は被験者それぞれに対し、一人ずつ他者とのコミュニケーション手段を断った上で 2 回おこなった。

1 回目の実験では 2 題の問題をそれぞれ「行動事例表示機能」をつかわずに解いてもらう。2 回目の実験では「行動事例表示機能」をつかいながら行動計画をしてもらう。1 回目の実験で蓄積された行動事例を、2 回目の実験の「行動事例表示機能」において用いる。そのため、被験者全員に対して 1 回目の実験を行ってから、2 回目の実験を行う。

問題 1、問題 2 を解く制限時間は 5 分と設定した。

6.3 実験結果

実験の結果、多くのユーザにとって理解しやすいと考えていた ToDo リストのインターフェイスについて、多くの勘違いがあった。通常の ToDo を追加していくという縦の追加方法に加え、ToDo を分解していくという横の追加方法があったため、適切に分解できないユーザが多かった。そのため十分な ToDo 分解例のサンプル数が集まらず、2 回目の「行動事例表示機能」の実験は行えなかった。

また、今回実験に協力してもらった村井研究室のメンバーは研究室のネットワーク管理に直接関わっておらず、先生にネットワークの相談を受けたときにどのように行動すれば良いかについて具体的な行動計画を全く行えないユーザが多かった。それに対して、コンピュータ利用相談員は日頃から相談を受けているために具体的に行動計画を行う事が出来た。

今回の実験の結果を下記の表にまとめた。

表 6.1: Shumu 検証の結果

所属	被験者	軸の理解	具体的 ToDo 数	グループ平均
村井研	A	順接／分解	0	1
	B	順接／分解	2	
	C	関連／関連	1	
利用相談員	D	関連／関連	5	5.3
	D	並列／順接	6	
	E	条件分岐／順接	5	

表 6.1の「軸の理解」とはユーザがどのように ToDo の展開をしていったか、その展開の軸をどのように解釈していたかを観察したものである。「具体的 ToDo 数」はユーザが問題 2「『ネットワークがつながらない』と先生が言ってきた」で具体的に ToDo を分解できた数を表している。「グループ平均」は「具体的 ToDo 数」のグループごとの平均値である。以下ではそれぞれの項目と結果について詳細に説明する。

まずは「軸の理解」について詳細に説明する。今回の実験ではユーザが、ToDo を縦に追加していく「縦の軸」と、ToDo を横に展開し細分化を深めていく「横の軸」の意味を勘違いしてしまう例が多かった。定型的な問題である問題 1 の「ハノイの塔」では横に展開する必要があまりなく、多くのユーザが縦に追加して郁子で順接を表していた。しかし、非定型的な問題である「ネットワークにつながらないという相談」の問題を分解し始めると多くのユーザが軸を勝手に解釈し始めた。その展開の仕方は 4 つあった。

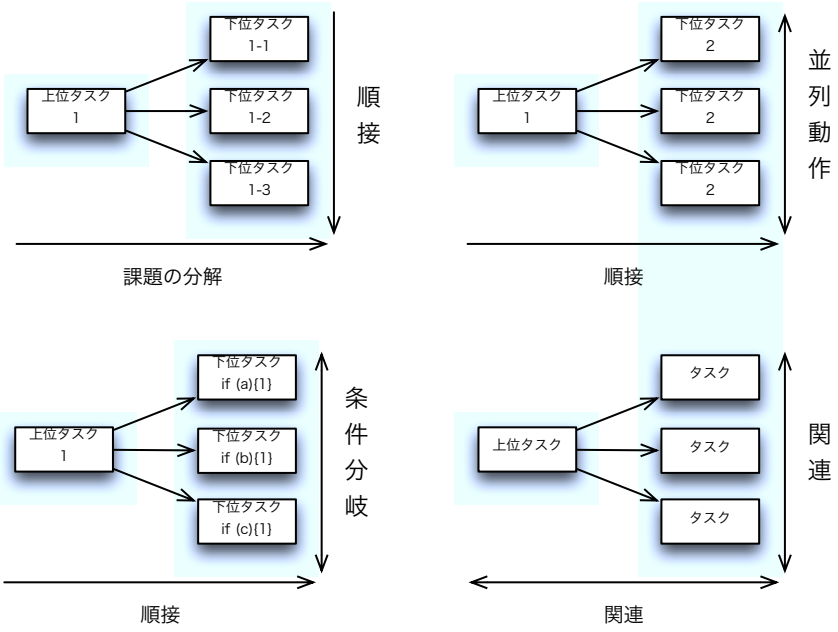


図 6.2: ユーザの ToDo 展開のパターン

図 6.2は Shumu の画面上でユーザが下と右に追加していくタスクをどのように位置づけていたのか、その追加の軸の意味のパターンを観察したものである。左上の図は縦の軸はタスクが「順接」、つまり上から順番に展開されていくと考え、横の軸はタスクを細分化していく「課題の分解」であると考えるパターンを示している。Shumu の設計が想定している思考パターンである。右上の「順接」／「並列動作」は左から右へタスクの処理が流れていき、並列でこなす事が出来るタスクの場合のみ表示するという考え方であった。左下の「順接」／「条件分岐」は基本的にタスクの処理は左から右に流れていくと考え、条件によって分かれる事もあると考えるパターンである。今回のネットワークの相談を受けた場合の対応について、問題点を明確にする判断チャートをつくろうとする被験者が見られた。右下の「関連」／「関連」は縦横の軸の意味を特に考えず、マインドマップのように関連するタスクを付け加えていくという方法をとっていた。

このように現在の Shumu のインターフェイスでは様々な軸の捉え方をされてしまうという問題がある事が分かった。

次に「具体的 ToDo 数」について説明する。表 6.1の「具体的 ToDo 数」は、非形式的問題である問題 2 の計画において、ユーザがどれだけ具体的な ToDo を記述する事が出来たか、という数を表している。この数は「先生からネットワークがつながらないという相談をうけた」というタスクを分解してもらった下位 ToDo の中から、「頑張って対応する」などといった具体的ではない ToDo を除いた数である。この表を見ると、コンピュータやネットワークについての知識をもつ村井研究室のメンバーでも、相談を受ける経験が少ない場合は具体的な ToDo に行動計画できない傾向がある事が見て取れる。つまり、ユーザがある課題を有効な ToDo に分解できるかどうかを見る事によって、ある程度そのユーザが課題に対して経験やノウハウを持っているかを計る事ができると考えられる。

6.4 考察

本検証の結果、「ToDo リストを見る事によって、ユーザがその ToDo の示す課題に対して経験を持っているかどうか」を調べる事が出来る可能性が示唆された。しかし、ToDo をツリー構造に分解していくというインターフェイスには様々な理解パターンが存在し、設計者が求めているようには ToDo データを入力してくれないという問題をはらんでいる事が分かった。今回の実験では問題の難易度が高く制限時間が短かったために「行動事例表示機能」で使える ToDo 分解事例が収集できず。仮説 1 および仮説 2 の検証はできなかった。

今後ユーザインターフェイスに対しては、ユーザが誤解しないようなインターフェイスを作り込んでいくことが求められる。もしくはユーザに Shumu を長い時間使ってもらい、その ToDo 分解の仕方について訓練を積んでもらう必要もあるだろう。ユーザは定型的な問題であるハノイの塔では分解の軸について混乱しておらず、非形式的な問題では軸の設定について混乱が見られたことから、問題が複雑であればあるほど自らの ToDo 分解において軸の混乱が見られるのだと思われる。そのような傾向があるのであれば、ユーザにもある程度 Shumu の ToDo 分解の仕方について訓練を積んでもらう必要があると考えられる。

実験についての反省を述べるならば、問題の解答時間をもう少し長くするべきであった。また、村井研究室における被験者も、利用に関しての相談をうける役割を担っている、ネットワーク管理者に頼むべきであった。

第7章　結論

7.1　まとめ

本研究はノウハウ探索における Web 検索の問題点を指摘し、その解決策を考えるなかで「人に聞く」ことまで支援範囲に含めたノウハウ探索支援システム「Shumu」を提案、設計した。Shumu によりユーザのノウハウ探索活動が改善され、獲得できるノウハウの量も幅も広がることが期待できる。

Web 検索は図書館における情報探索をメタファーにした物であり [4]、ユーザの抱えているタスクは切り離されて考えられる。[5] そしてユーザは欲しいと思った情報を具体的に思い描き「検索する」という行為を明示的にとらなければならない。1 つ目の問題は、ユーザが行動したときに初めて情報がユーザに提供される「プル型」のシステムでは、ユーザが情報の存在を信じて行動を起こさなければユーザと情報が出会う事はないということであった。Shumu ではユーザの入力する ToDo データから常にユーザが抱えているタスクに対応したノウハウを検索しており、ユーザが探索行動を起こさないことで生まれる機会損失を抑える事が出来る。

２つ目の問題は Web 検索では言語化されたノウハウしか検索する事は出来ないということであった。検索対象とする物をコンピュータが直接収集し、インデックス化する必要があるためである。しかし、コンピュータはまだ脳内の情報にアクセスする事は出来ないため、未言語化ノウハウを直接検索対象にする事は出来ない。Shumu では、完了した ToDo はユーザの持っている「経験」を指すインデックスであると考え、完了した ToDo を検索対象にすることで人の脳内に蓄積された「経験」を指し示すことができるとし、さらに間接的に、脳内に潜在する未言語化ノウハウの存在を指すインデックスとして、人にノウハウの存在を気付かせる事が出来るのではないかと考えた。

3 つ目の問題はユーザが背景知識を持っていないと正しい欲求認識を出来ず、本当の目的に対して合理的な行動を行うことができないということであった。Shumu は ToDo をツリー構造で表す事で常に行動の裏に潜む「本当の目的」を考慮するようにしている。また、そのような背景知識を持っていない行動を起こすときにはやはり「人に聞く」という行動が最も適切であると考え、それをサポートすることも Shumu では目指している。

Shumu は ToDo を社会的距離が近い他者から検索できるようにした。ノウハウ探索における社会的コンテクストは社会的距離が近いほど共有されており、社会的距離が近いユーザによる Todo の分解例の方が、ユーザにとってより有用であると考えられるためである。また、「誰に聞けば良いか」もしくは「誰に頼めば良いか」といった事が分かれば、その後の「人に聞く」や「人に頼む」といった行動を起こしやすくすることができ、今までは活用されてこなかった近隣のユーザのもつノウハウも有効に活用する事ができる。

これまでに説明してきた Shumu の方法論を検証するために、被験者にユーザとなって

もらって実際に利用してもらった。残念ながら現状 Shumu のユーザインターフェイスでは ToDo の入力方法がうまく理解されず、有効な解答パターンがあまり回収できなかったために Shumu の「行動事例表示機能」については検証を行えなかった。しかし、与えられた課題の、「ToDo 分解の精緻度」と「経験」は相関が見られ、ToDo が「経験」を示していると考える Shumu のアプローチを部分的に支持した。

Shumu の方法論をさらに検証し、社会的距離の近い人間が脳内に潜在的に持つノウハウを検索することが出来るようになれば、今まで活用されてこなかった周りの人間のノウハウや人的リソースが有効に活用され、社会厚生の増加に貢献する事が出来る。

7.2 今後の課題

本研究の今後の課題や方向性について述べる。

7.2.1 ユーザインターフェイスの改善

実験で述べた通り、Shumu の現在のインターフェイスでは ToDo の追加の意味が曖昧で、ユーザが勝手に定義してしまう傾向がある事が分かった。その問題を改善するためには軸の意味を勘違いしないようなインターフェイスを作り込む必要がある。Shumu 自体は ToDo の内容や依存関係を認識していないので、ユーザの入力を改善するしか方法はない。

リストの下への追加は「順接」であり、右への展開は「分解」であることを常にメッセージとして表示したり、アニメーション効果を加える事で気付かせたりすることがまずは考えられる。良い ToDo データのサンプルが集まり、「行動事例表示機能」が効果的に機能し始めた場合、ユーザは自然に ToDo の展開の意味を理解するようになっていくことも考えられる。ToDo リストの展開の仕方も、多くの ToDo リストを見る事で、人は模倣する事が出来るだろう。

もしくは、ToDo リストをテキストで記述するという普遍的なデータ形式をあきらめ限られた問題に特化することで、ユーザの操作可能範囲を絞ったユーザインターフェイスを設計する事が出来る。例えば、料理の手順のみを共有するシステムであれば、「下ごしらえ」や「盛りつけ」などのいくつかのステップ分けに関するひな形が提供できるであるだろう。問題範囲を限定する事でユーザにとってより分かりやすい物にすることができる可能性がある。

7.2.2 より正確で大規模な検証

本研究では「行動事例表示機能」の検証を行う事が出来なかった。この機能を検証するためには、多くのユーザに対して利用してもらい、ToDo 分解事例を蓄積していかなければならない。また、社会的距離が近いユーザのノウハウを探索するという仮説 2 を検証するためには、複数の社会集団に対して適切な問題を設定する必要がある。このような検証を通して、Shumu の方法論に対してより一層の検証を加えていく事が求められる。

7.2.3 マッチングアルゴリズムの高度化

本研究における Shumu のスコアリングは単純に ToDo を記述したテキストの単語の一致数で比較するというものであった。Shumu はあくまでもユーザに参考になりそうな事例を表示する事を目的としているので、多少不正確であっても問題はなく、最後の判断は人間が行うべきであると考えたためである。

知識処理の研究分野では、タスクをより厳密に定義してユーザの抱えているタスクに対して対応方法を提示する「エクスパートシステム」のような推論エンジンがある。[15] このようなシステムではタスクをより厳密に定義するために、タスクを記述する語彙を「タスクオントロジー」として整理する手法が研究されている。[12] Shumu でも知識処理の分野で培われたタスクオントロジーの扱い方などのノウハウを用いる事で、ToDo の近似度の計算精度を上げる事ができると考えられる。

具体的にはユーザが入力した ToDo のテキストを名詞や動詞ごとに分解し、同じような意味で使われていそうな単語のシソーラスを構築する。スコアリングのときにシソーラスを参照し、ユーザの ToDo テキストを語彙的に展開してマッチングする事で、同じタスクを意味する ToDo をより多く見つける事が出来ると考えられる。

また、現状では ToDo 事例をユーザが評価する仕組みが組み込まれていないが、ユーザに参考になった ToDo 事例には「参考になった」と評価してもらう事で、より多くの「参考になった」という評価を得た ToDo に優先的に高いスコアリングをするアルゴリズムなども考えられる。

7.2.4 他のシステムとのハイブリッドモデルの構築

ノウハウの共有・再利用を促進するためシステムとして Q&A システムがある。これはユーザが質問を投稿すると、別のユーザがその質問に答えるという電子掲示板を発展させたシステムである。このシステムを使うと一度行われた質問は明示化され検索対象になるため、次からは問題とそれに対する解答であるノウハウが検索可能になる。ノウハウ全体の集合において、検索可能な言語化された部分を増やしていく事が出来るシステムである。

この Q&A システムは、「誰がどのようなノウハウを持っているか」を検索する Shumu と補完的な役割を持っている。Shumu の上で質問とその解答を蓄積する事が出来れば、その質問とユーザの抱えているタスクを対応させて記録する事が出来る。つまり、Shumu の行動事例表示機能の延長戦上で同時に Q&A システムの検索も行う事が出来るようになる。

その他の情報共有システムも Shumu の ToDo と対応させて記憶させる事で、ユーザが抱えている ToDo にあった情報を表示させられる可能性がある。

謝辞

本論文の作成にあたり、ご指導いただいた慶應義塾大学環境情報学部教授の村井純博士、徳田英幸博士、中村修博士、武田圭史博士、同学部准教授の楠本博之博士、高汐一紀博士、植原啓介博士、三次仁博士、同学部専任講師の重近範行博士、Rodney D. Van Meter III 博士、中澤仁博士に感謝いたします。また、絶えずご指導とご助言をいただきました慶應義塾大学大学院政策・メディア研究科特別研究講師の斉藤賢爾博士、に深く感謝いたします。研究の進捗が非常に悪く、期限を過ぎてもマイルストーンを達成できない事が多かった私を最後まで見捨てず、辛抱強く見守ってくださいました。そして本研究を進めていく上で、様々な励ましと助言、お手伝いをいただきました、慶應義塾大学徳田・村井・楠本・中村・高汐・重近・バンミーター・植原・三次・中澤・武田合同研究プロジェクトの皆さまに感謝いたします。私は経済学部ながら情報科学を学ぶことを希望し、湘南藤沢キャンパスの村井研究室の門戸を叩きました。そのようなわがままを許し、私に学ぶ機会をくださった皆様には重ねて御礼申し上げます。村井研の NECO 研究グループのメンバーと三田キャンパスのコンピュータ利用相談員の皆様は Shumu を実際に使ってくださり、実験に協力してくださいました。大変有意義なアドバイスもいただけました。ありがとうございました。共に大学に何泊もし、一緒に卒論を書いた総合政策学部 4 年の峯岸宗弘君からは多くのアドバイスと励ましをいただきました。ありがとうございました。最後に、研究期間中夜遅くまで起きている私を見守ってくれた母に心から感謝いたします。

関連図書

[1] 「質問できない新人」の簡単な指導法.

[2] なぜ新人は聞きに来ないのか? - teruyastar はかく語りき.

[3] プログラマで、生きている: ググるな危険.

[4] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, Vol. 30, No. 1-7, pp. 107 – 117, 1998. Proceedings of the Seventh International World Wide Web Conference.

[5] Peter Ingwersen and Kalervo JSrvelin. 情報検索の認知的転回 情報捜索と情報検索の統合. 丸善, 1 2008.

[6] Michael R. Solomon. *Consumer Behavior (7th Edition)*. Prentice Hall, 7 edition, 3 2006.

[7] フリー百科事典『ウィキペディア（Wikipedia）』. Aidma.

[8] フリー百科事典『ウィキペディア（Wikipedia）』. コンテキストアウェアネス.

[9] フリー百科事典『ウィキペディア（Wikipedia）』. 手続き的知識.

[10] マイケルポランニー. 暗黙知の次元 (ちくま学芸文庫). 筑摩書房, 12 2003.

[11] 佐藤一夫, 山本真人, 小林功, 佐治信之, 田中克明. 行動履歴に基づく情報推薦基盤と推論エンジンの開発 (「web インテリジェンス」及び一般). 電子情報通信学会技術研究報告. AI, 人工知能と知識処理, Vol. 108, No. 119, pp. 33–38, 20080623.

[12] 石川誠一, 久保成毅, 古崎晃司, 來村徳信, 溝口理一郎. タスク・ドメインロールに基づくオントロジー構築ガイドシステムの設計と開発 : 石油精製プラントを例として. 人工知能学会論文誌 = Transactions of the Japanese Society for Artificial Intelligence : AI, Vol. 17, pp. 585–597, 20021101.

[13] 杉本徹雄. 消費者理解のための心理学. 福村出版, 6 1997.

[14] みずほ情報総研株式会社 吉川日出行. サーチアーキテクチャ 「さがす」の情報科学. ソフトバンククリエイティブ, 9 2007.

[15] 溝口理一郎. 知識の共有と再利用研究の現状と動向 (【特集】「知識の共有と再利用」). 人工知能学会誌, Vol. 9, No. 1, pp. 3–9, 19940101.