平成18年度

筑波大学第三学群情報学類

卒業研究論文

題目
webページにおける閲覧者間の繋がりを用いたインフォーマルコミュニケーション支援

主専攻　　情報科学主専攻

著者　　佐藤　俊輔

指導教員　田中二郎　高橋伸　三末和男　志築文太郎

要　旨

コミュニケーションには人と人との繋がりが重要である。近年 Blog や SNS などの特にインフォーマルコミュニケーションを支援するサービスが脚光を浴びているが、これらは「同じ趣味を持っている」や「友達の友達」といった繋がりが根底にある。本研究では「偶然同じ Web ページを見ている」という繋がりをユーザに知らせることでインフォーマルなコミュニケーションのきっかけを作り出し、そのきっかけを活かし円滑にコミュニケーションができるチャットシステムを開発する。またその発展として繋がりを積極的に作り出す機能であるホットページランキングを導入した。

目 次

図 目 次

第1章　序論

1.1　背景

近年、Blog や趣味などを公開することで知人の和を広げていく mixi[1]が２０代から３０代の世代を中心に関心を集めている。また mixi はソーシャルネットワーキングサービス (SNS) に分類される。Blog・SNS の言葉の定義と特徴は 1.2.1 節で述べるが、2006 年 3 月末時点で blog の利用者数（登録者数の合計）が 868 万人、SNS は 716 万人である[2]。情報発信やコミュニティ形成を効率的に行うためのツールとして発展してきた両者だが、目的は既述の通り情報発信とコミュニティ形成であり、その後発生するコミュニティ内でのコミュニケーションである。利用者数の増加傾向から今後も Blog・SNS などのネットワークコミュニケーションを支援するサービスは大きい需要があると予想される。
コミュニケーションは TPO に応じて使い分けられるフォーマルなものとインフォーマルなものがあるが Blog・SNS で行われるコミュニケーションはインフォーマルなものが多い。井戸端会議・雑談に代表されるインフォーマルコミュニケーションは会議などフォーマルなコミュニケーションと比較すると，偶発的で相手も話題も不特定という特徴がある [2]。その特徴故にインフォーマルコミュニケーションの中ではしばしば柔軟なアイディアが生まれることもあり、インフォーマルコミュニケーションは豊かなライフスタイルの形成に有益であると考えられる。しかし現在のコミュニケーション支援サービスにはいくつかの問題点がある。その問題点については 1.2 節で述べる。本研究ではその問題点を改善したインフォーマルコミュニケーションを支援するシステムを提案する。

1.2　コミュニケーション支援サービス

1.2.1　Blog・SNS に見るきっかけとしての「繋がり」

- IT 用語辞典[3] によれば Blog とは『個人や数人のグループで運営され、日々更新される日記的な Web サイトの総称。内容としては時事ニュースや専門的トピックスに関して自らの専門や立場に根ざした分析や意見を表明したり、他のサイトの著者と議論したりする形式が多く、従来からある単なる日記サイト (著者の行動記録や身辺雑記) とは区別されることが多い。』である。

[1]mixi:http://mixi.jp/
[2]総務省：http://www.soumu.go.jp/
[3]IT 用語辞典：http://e-words.jp/

Blog の特徴としてトラックバック機能を利用することで、記事がクチコミで広まりやすく多くの人を集めることができるということが挙げられる。同じサイトに対してリンクを張った者同士、または自分の記事に興味をもちリンクを張ってくれた人達などの「繋がり感」がコミュニケーションを促すきっかけとなっていると考えられる。

- IT 用語辞典によれば SNS とは『人と人とのつながりを促進・サポートする、コミュニティ型の Web サイト。友人・知人間のコミュニケーションを円滑にする手段や場を提供したり、趣味や嗜好、居住地域、出身校、あるいは「友人の友人」といったつながりを通じて新たな人間関係を構築する場を提供する、会員制のサービス』である。
  SNS の特徴として、SNS の中では FOAF（Friend Of A Friend)、すなわち「友人の友人」が重要になっている。FOAF は自分の個人情報などのプロフィール（本名、ニックネーム、職業、居住地、誕生日、趣味、関心、友人関係など）を記述し、相互の「繋がり」を可視化・共有することでコミュニケーションのきっかけを作り出している [4]。

1.2.2 既存の「繋がり」の問題点と改善案

Blog では「興味の対象が同じである」という繋がり、SNS では「FOAF」という繋がりを支援していて、コミュニケーションのきっかけを作り出している。そこで「繋がり」がインフォーマルコミュニケーション生起における重要なファクターであると考えられる。しかし上記のサービスにおいて新たな繋がりが構築されるためには、トラックバックを張ったり、FOAF のページにアクセスしたりと、ある程度能動的な繋がり探しをしなければならない。これらのプロセスは積極的な活動として精神活動に有益であるが、裏を返せば「繋がり」を**探す**という非日常的活動を行わなければ「繋がり」は得られないということである。
本研究では既存のサービスの良さである「繋がり」探しを継承する一方で、本人と不特定多数の他ユーザとの「繋がり」を**探さずに**日常的活動の中から構築することで、インフォーマルコミュニケーションのきっかけを提示することを目指す。

1.2.3 既存のコミュニケーションツールの問題点と改善案

インターネットを用いてコミュニケーションを行うツールには、掲示板やチャット、Blog のコメント欄等、様々な種類がある。これら既存のコミュニケーションツールにおいて、自分が話したいテーマに沿った掲示板やチャットルームを探したりするプロセスが必要となる。それらの能動的なアクションは、会話へのモチベーションがある程度高くなければ起こしにくい。さらに本研究はインフォーマルコミュニケーションに焦点を当てていて、その特徴の偶発的 [1] という点に着目するならば突発的に沸いたモチベーションを下げさせないために会話は相手と即時的に行われる方が好ましい。そのため B. A. Nardi[3] らが議論しているレスポンスの即時性・明確な相手との接触を持てるという利点から、使用するコミュニケーションツールはチャットシステムを採用する。しかし既に述べたように既存のチャットツールでは部屋を探すというプロセスを踏まなければならないため、本研究が必要とする即時性は備えていない。

そこで本研究では話す場所を探すというプロセスを排除し、その場で会話を行えるチャットツールを開発する。

1.3 論文の構成

1 章で Blog や SNS がどのような特徴をもつサービスであり、どのような問題点があるのかを述べた。インフォーマルコミュニケーション支援に必要な「繋がり」の提示とそれを活かすシステムの提案、及び利用例を 2 章で述べる。3 章ではシステムの機能と動作例を説明し、4 章で実装方法を述べる。
5 章で本システムに関する考察を行い、6 章で関連研究との位置づけを述べた後に 7 章でまとめる。

第2章　インフォーマルコミュニケーションを誘発させるシステムの提案

2.1　提案システム：Page-Chat

本研究では 1.2.2 節と 1.2.3 節を受け次に挙げる 3 つの要件を満たすシステム Page-Chat を提案する。

2.1.1　日常的な活動における繋がりの構築

Page-Chat ではユーザの日常的行為の一つとして Web ページの閲覧に注目する。Web ページは実世界での場所と同じ意味を持ち、偶然同時刻に同じページを閲覧しているユーザ相互に「繋がり感」を感じさせることができる。このメリットは以下の 2 つである。

- **探さなくても**「繋がり」が構築できる
- 「自分と同じ記事に興味を持つ人間がいる」という親近感と記事の話題性がコミュニケーションのきっかけとなる。

2.1.2　積極的な繋がり探索

2.1.1 節の偶然同じ Web ページを見ているという繋がりは Web ブラウジングをしていた場合の副次的な産物として感じることができるが、この同じ Web ページを見ているという状況を積極的に作り出すことで更にネットワークコミュニケーションは豊かになると考えられる。そこで Page-Chat は多くの人が閲覧している人気ページの存在をユーザに気付かせる機能を実装する。この機能のメリットは以下の 2 点である。

- 積極的に他者との繋がりを構築することができる。
- 未だ見ぬ人気ページのコンテンツを試しに閲覧することで、新しく興味を引かれる記事に出会える可能性がある。

2.1.3 即時的な会話

インフォーマルコミュニケーションは「偶発的で話題も不特定である」という特徴 [2] のためにきっかけはあっても、しばしばコミュニケーションが発生しないケースがある。例えば隣に人がいるときだと、話したいと思えば隣人と会話が気軽にできる。しかしそうでない場合は、既存のコミュニケーションツールでは突発的に沸いたこの会話欲を満たす空間はそのトピックスに関する掲示板やチャットルームになる。この時都合よくテーマに沿った掲示板・チャットルームを探したりするプロセスは、突発的に沸いた会話欲に対する熱を冷ましてしまう恐れがある。
より円滑なコミュニケーションを支援するために Page-Chat では『場所を探すことなく』閲覧者同士で即時的な会話を行えるようにする。そのために閲覧中の Web ページを一つのチャットルームと見なして、その場で話せるチャットシステムを開発する。

2.2 利用例

提案するシステムの利用シーンを二つ挙げる。

2.2.1 偶発的な繋がり

A 氏は釣りが好きで休日はよく各地の釣りスポットへ出かける。たまには友人達と一緒に行くこともあるが、友人達は A 氏ほど釣りに深く関心を持っているわけではないのでほとんどの場合は A 氏一人で行く。
いつものように今度の休日に出かける釣りスポットを探すために、自宅のパソコンで釣りスポット紹介サイトを見ていた。各スポットの風景や釣れる魚の種類を慎重に吟味するが、実際に行ってみないと分からない事も多々あった。とりあえず気になったスポットに行ってみようかと考えていた時に**偶然同じサイトを見ている人がいる事に気付いた**。スポットについて何か情報が聞けるかもしれないので A 氏はチャットで彼に話しかけてみることにした。話しかけられた B 氏は釣り好きがいたことに喜び、自分が行ったことのあるスポットの感想・今度行こうかと思っているスポットなどを話してくれた。A 氏と B 氏は存分に釣り談議を楽しんだ。

これは同じ興味を持っているユーザ同士が偶然居合わせ、コミュニケーションが発生した例である。A 氏は掲示板に書き込むほど迷っていたわけではなかったので、コミュニケーションが発生する余地は無かった。しかし同じページに人がいると分かると、会話欲が沸いてきて B 氏と雑談で盛り上がった。
繋がりを知らせてその場で会話できる Page-Chat の典型的な利用例である。

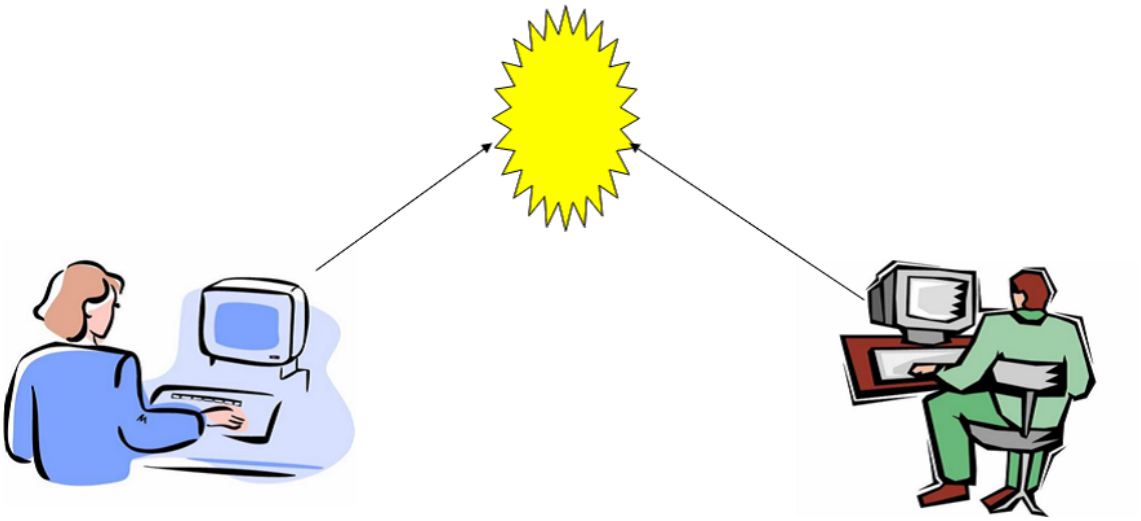


図 2.1: 偶然同じ Web ページを見ている

2.2.2　積極的な働きかけ

釣りスポット探しが大成功に終わった A 氏はブラウザを閉じようとしたが、何やら**人が多く集まっているページがある事に気付いた**。好奇心を抱いた A 氏はそのページを閲覧してみることにした。
そのページはニュースサイトでサッカー日本代表の試合速報が報じられていた。サッカー強国ブラジル代表を相手に前半戦は日本代表がリードしている、といった内容の記事でチャットの会話を見ると日本の選手達が一丸となって失点を防ぐ様を称える内容だった。もうすぐ後半戦が始まるらしい。サッカーは詳しくなかったが皆の興奮ぶりに興味を引かれた A 氏はテレビをつけた。

これは人気ページの存在を知り、新たな記事に対して興味を持つ例である。A 氏は記事の内容に対して詳しくなかったが、今後興味を持つことでそのジャンルにおけるコミュニケーション生起の可能性を示唆している。また A 氏が詳しい内容の記事であったならばその人気ページで一緒に盛り上がった可能性もある。

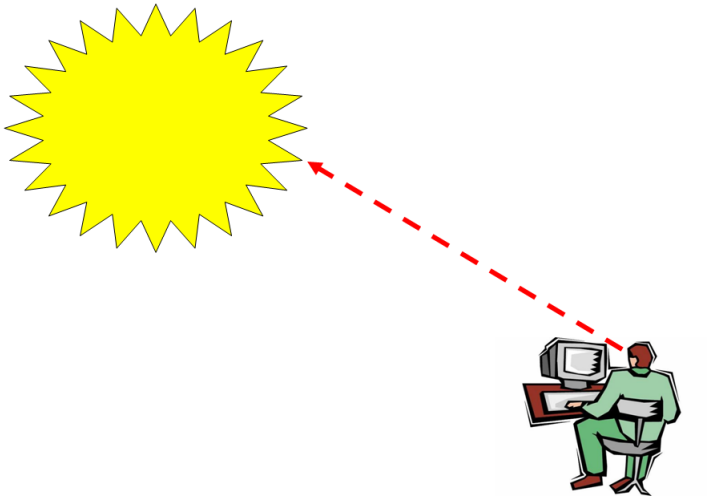


図 2.2: 人気があるページに気付く

第3章　Page-Chat

この章では Page-Chat のクライアントアプリケーションとその動作例を説明する。

3.1　クライアントアプリケーションの説明

図 3.1 は Page-Chat の実行画面である。

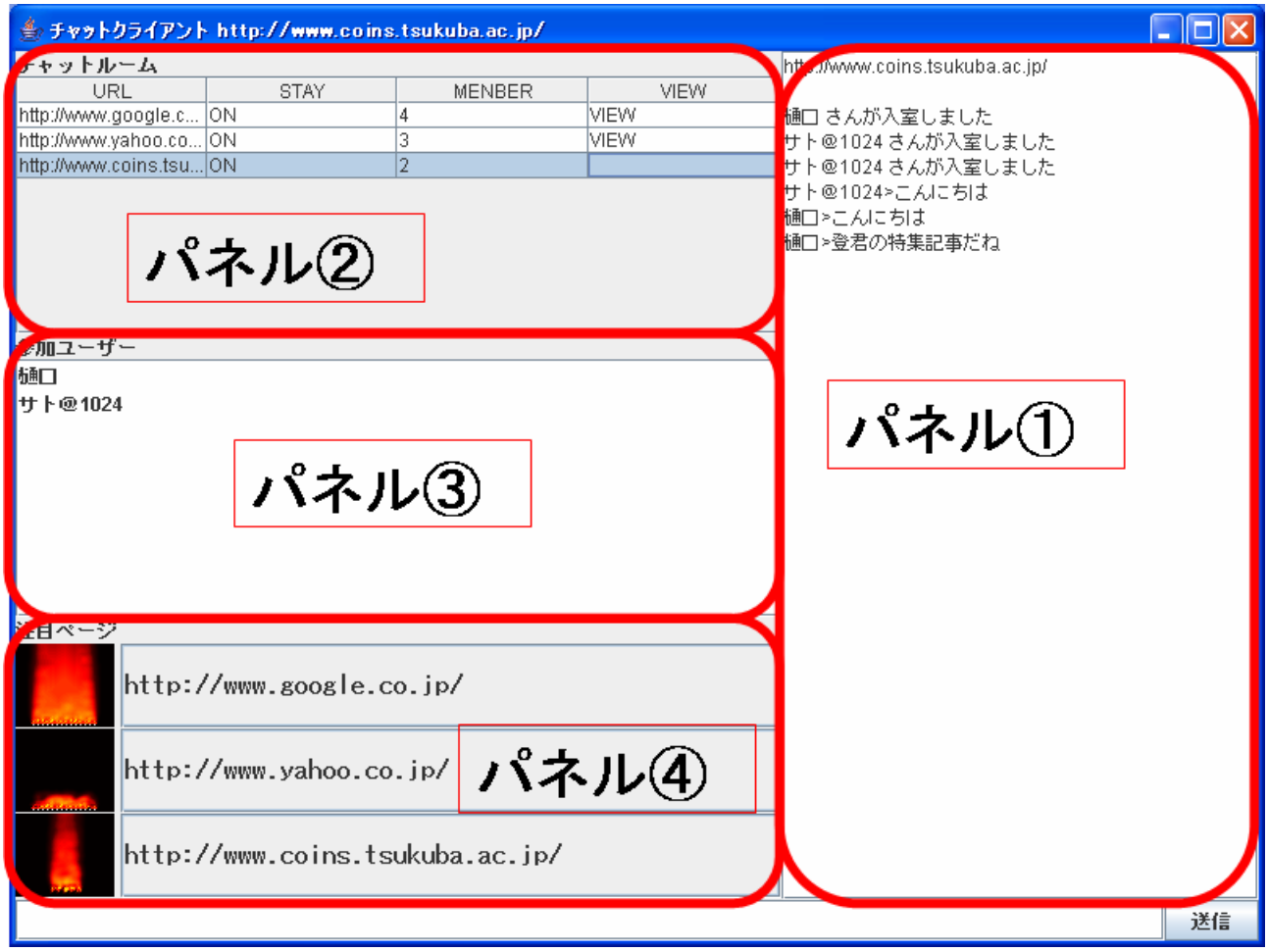
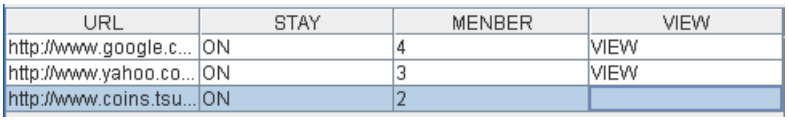


図 3.1: PageChat 実行画面

3.1.1　パネル①：会話スペース

利用者間で会話を表示するためのスペースである。スペースの最上段の行は会話をしているウェブページの URL である。これは通常のチャットアプリケーションでの部屋名にあたる。

3.1.2　パネル②：ルームリスト



URL	STAY	MENBER	VIEW
http://www.google.c...	ON	4	VIEW
http://www.yahoo.co...	ON	3	VIEW
http://www.coins.tsu...	ON	2	

図 3.2: ルームリスト

部屋名 (URL) を行別にリスト化したものであり、現在入室中の部屋を表示している。左から順に URL,STAY,MEMBER,VIEW の列があり、それぞれ各部屋の状態を示している。またチャットを快適に行うためのアクションを起こす時に用いる。

[URL]
ブラウザでウェブページを訪問するとそのページの URL 名の部屋に自動的に入室する。ルームリストにはユーザが自ら訪れたページの部屋のみを表示する。

[STAY]
STAY セルの初期状態 (入室時) は「OFF」にセットされている。この状態で新規に別のページを閲覧すると自動的に以前のページの部屋から退室する。STAY セルをクリックするとセルの値は「ON」にセットされる。この状態で新規に別のページを閲覧したとしても以前のページの部屋から退室することはない。これにより大量の訪問済みのページ (部屋) から自分の興味のあるページ (部屋) だけを残すことができる。また部屋に居続けることで新規にページを訪れた他ユーザとの出会いを誘発することができる。そのページに対する興味が薄れて退室したい時には、もう一度 STAY セルをクリックすることで即座に退室できる。

[MEMBER]
そのページ (部屋) にいるユーザ数を表示している。

[VIEW]
どの部屋の会話内容を会話スペースに表示するかを選択するものである。以前訪れた部屋が「STAY」状態であったとしても、新規に別のページを閲覧すると会話スペースには新規に訪れたページ (部屋) の会話内容が自動的に表示される。この時に任意の「STAY」状態の部屋の VIEW セルをクリックすることで、その部屋の会話内容が会話スペースに表示される。また VIEW セルの値は通常何もセットされていないが、現在会話スペースに会話内容を表示して

いる部屋以外の部屋で他ユーザによる発言等のアクションが起きたときは、対応する部屋の VIEW セルの値に「VIEW」がセットされる。ユーザは「VIEW」を見ることでどの部屋でアクションが起こったかを知ることができ、VIEW セルをクリックすることで即座にその部屋で起こったアクションの確認ができる。一度「VIEW」状態のセルをクリックするとそのセルは再び何もセットされていない状態に戻る。

3.1.3 パネル③：ユーザリスト

現在会話スペースで表示されている部屋の入室者一覧である。

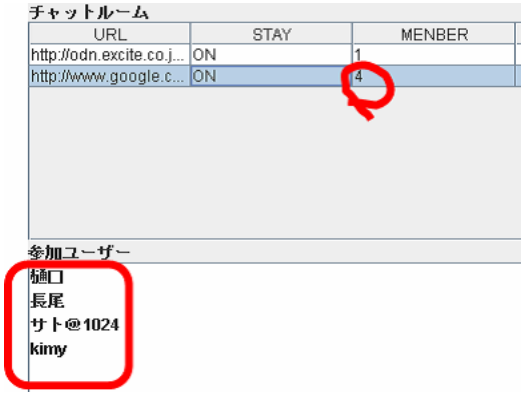


図 3.3: ルームリスト

3.1.4 パネル④：ホットページランキング

現在人が集まっているページ、つまり人気のあるページの URL を表示し、盛況具合を炎でアンビエントに表現する。今回の実装では、人気度は存在している全ての部屋のユーザ数を比較することで選出している。ユーザ数が多ければ多いほど炎は燃え盛ることになる。

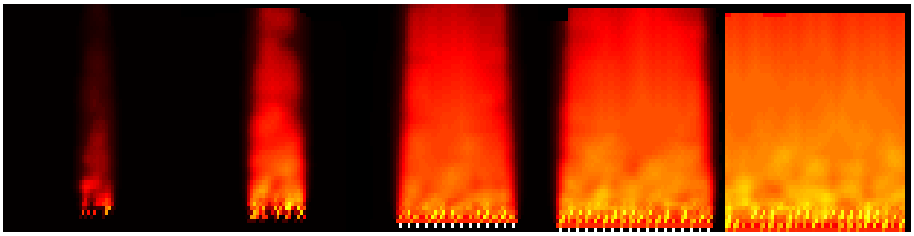


図 3.4: 炎レベル, 左からレベル 1〜レベル 5

3.2　動作例

Page-Chat の動作例を挙げる。

STAY セルの ON・OFF

(A) は google のページを見ているが STAY セルの値は OFF 状態になっている。この状態で新たに yahoo のページをブラウザで見るとルームリストから google が消え、yahoo のみが残る (B)。(B) の状態から yahoo の STAY セルを一回クリックして ON 状態にしておくと次に再び google を訪れた時、yahoo は残りルームリストには google が新たに追加される。(C)

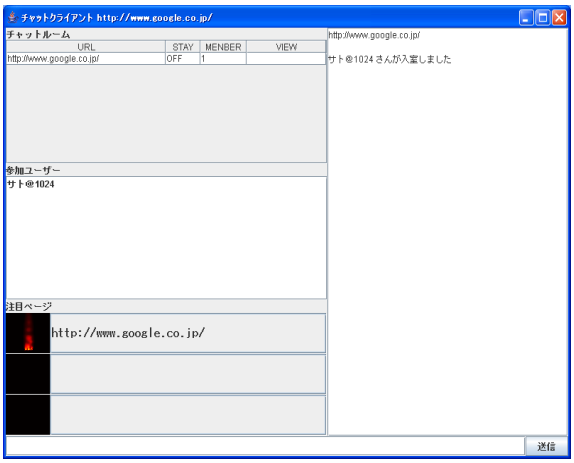


図 3.5: STAY セルの動作例 (A)

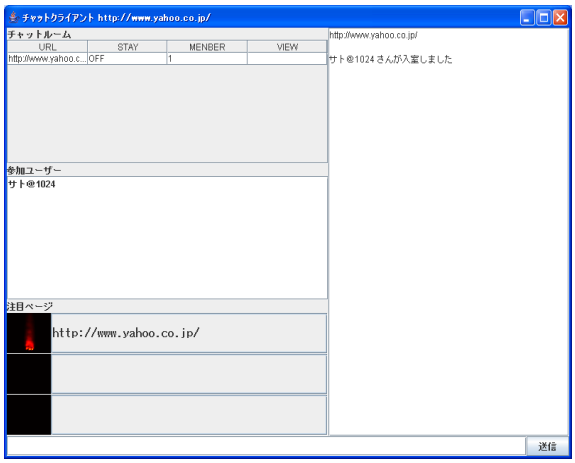


図 3.6: STAY セルの動作例 (B)

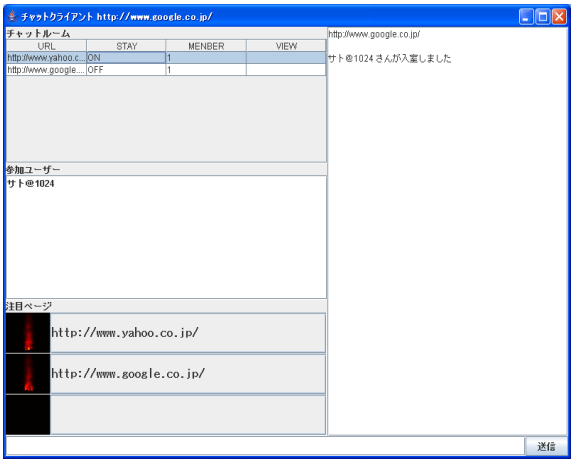


図 3.7: STAY セルの動作例 (C)

VIEW セル

(D) と (E) を見ると yahoo と google の両方の部屋に 2 人のユーザが入室していることが分かる。次に google 部屋を見ていると yahoo 部屋の VIEW セルの値が「VIEW」に変わったことに気付いた (F)。そこで yahoo 部屋の VIEW セルをクリックして覗いてみると yahoo 部屋で発言があったことが分かる。

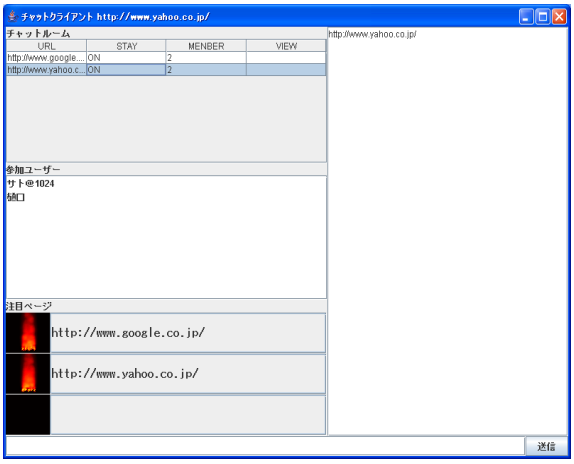


図 3.8: VIEW セルの動作例 (D)

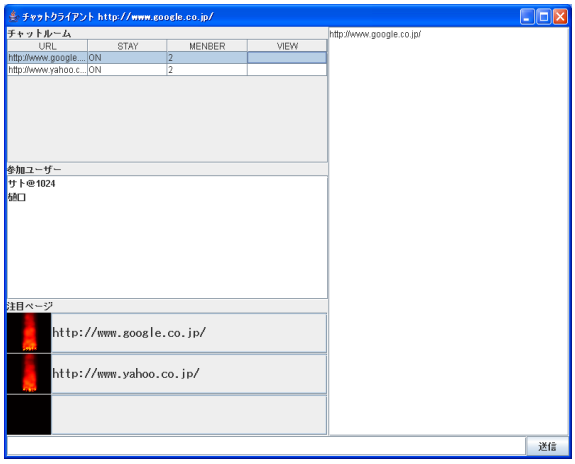


図 3.9: VIEW セルの動作例 (E)

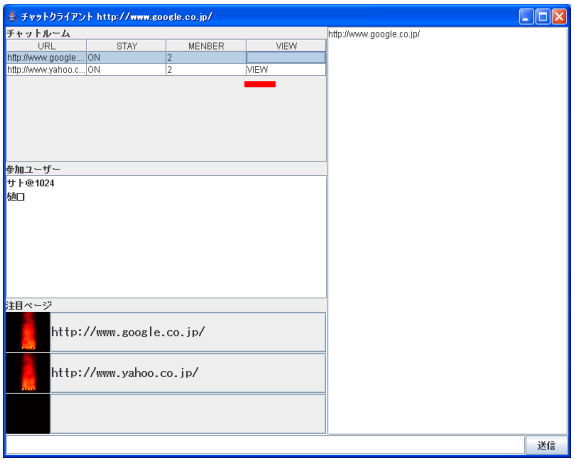


図 3.10: VIEW セルの動作例 (F)

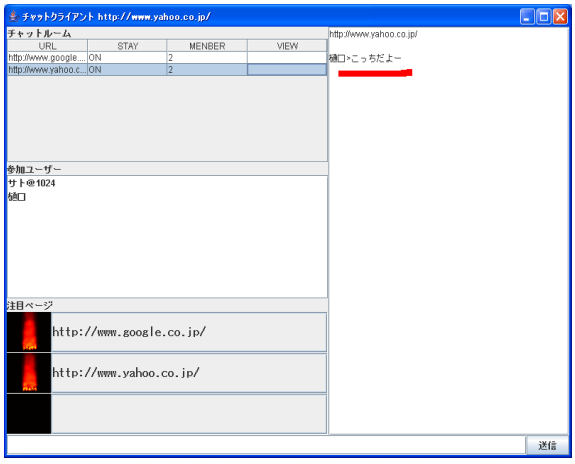


図 3.11: VIEW セルの動作例 (G)

第4章　システム構成

4.1　全体の構成

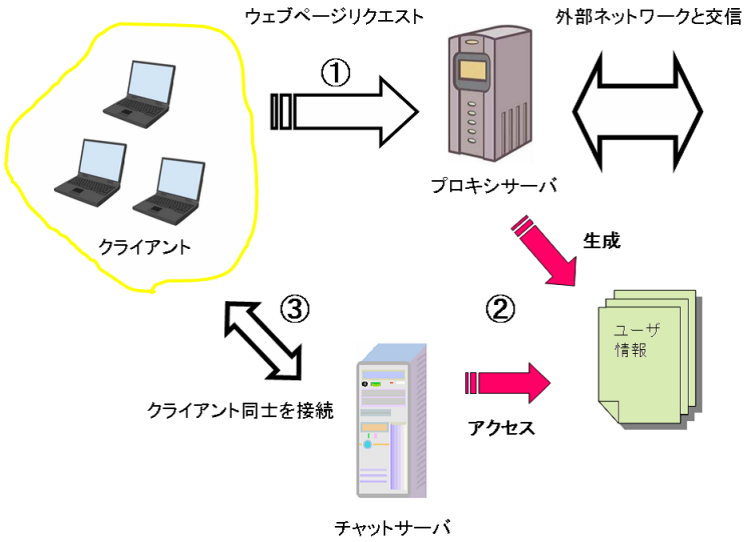


図 4.1: システム概観

図 4.1 はシステムの概観を表している。
プロキシサーバが得て保存したユーザの情報を用いてチャットサーバはチャット機能を提供している。

① チャットクライアントを起動しているユーザは好みのブラウザでウェブページを閲覧する。この時コンテンツのリクエストはプロキシサーバを介して行う。

② プロキシサーバはこのユーザの情報 (IP アドレス・リクエストライン) を取得してログファイルに保存する。この情報を元にチャットサーバはユーザの認証、チャットルームの作成・入室などの動作を行う。

③ チャットクライアントとチャットサーバが会話やホットページランキングなどのデータのやり取りを行う。

4.2　プロキシサーバの実装

開発環境は以下の通りである。また本研究におけるプログラムは全て同じ環境下で実装した

- CPU：Pentium(R)4:3.00GHz,Memory:1GB
- OS：Microsoft Windows XP Professional Version 2002 Service Pack 2
- 開発言語：Java(JDK5.0)
- 開発環境：Eclipse SDK3.1.1

4.2.1　概要

ほとんどの場合サーバとクライアント (ブラウザ) のプロトコロルは HTTP であり、実装を行う上でもこれを基本として進める。HTTP は大別するとリクエストとレスポンスで構成されている (図 4.2)。クライアントはサーバに向けてリクエストを送信する。リクエストの一行目は「メソッド　URI　HTTP バージョン」で構成されていて、URI で指定されている対象をどうメソッドで処理するのかが既述されている。これを一般にリクエストラインと呼ぶが、本研究でプロキシサーバを設置する目的はチャットクライアントアプリケーションを起動しているユーザのリクエストラインを解析することで、ユーザが見ているウェブページの情報を取得することである。またプロキシサーバで管理することによりブラウザに依存することなく情報が取得できる。

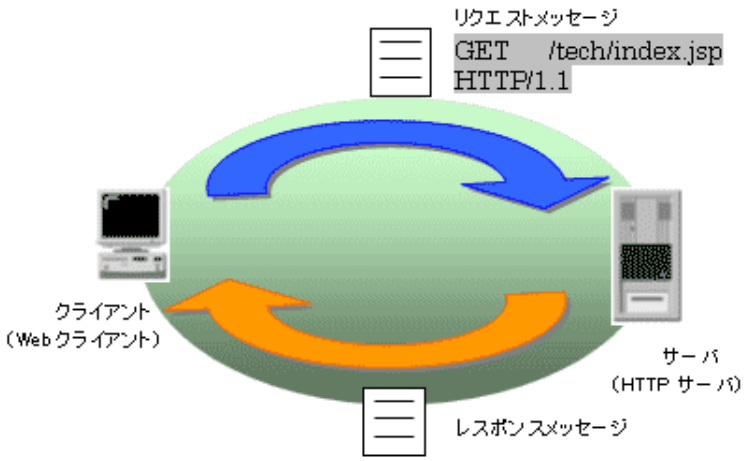


図 4.2: サーバとクライアントの関係

4.2.2　リクエストラインの解析

リクエストラインは「メソッド　リクエスト URI　 HTTP バージョン」で構成されている。ユーザがウェブページを 1 つリクエストするだけでもプロキシが処理するリクエスト文 (図 4.3) は膨大な量になる。その膨大な量のリクエスト文を解析してログファイル (4.2.3) に必要な URL を抽出する必要がある。
現段階では GET メソッドでフィルタリングをした結果を利用している (図 4.4)。フィルタリングした結果を時系列順で見てみると、先頭で HTML ドキュメントの URI(URL) をリクエストして以降の行では HTML ドキュメントに含まれる画像の URI をリクエストしている。全てのページのリクエストでこの法則が成り立っているかどうかの検証は行っていないが、現段階ではこの先頭 GET メソッドの URI をログファイルに書き込んでいる。

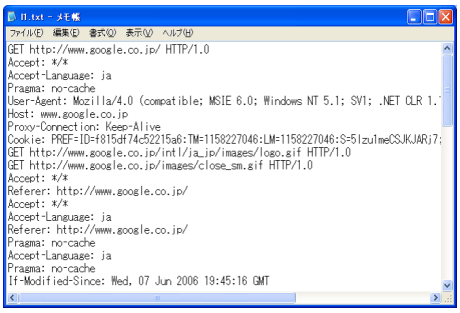


図 4.3: google のリクエスト文

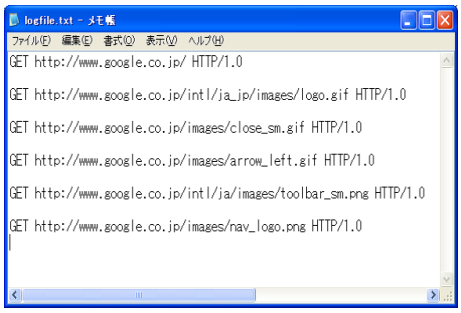


図 4.4: リクエスト文を GET メソッドでフィルタリングしたもの

4.2.3　ログファイル

必要なユーザの情報を保存するファイルである。ここでいう必要な情報とは「どのユーザ (IP アドレス) がどのページ (URL) を見ているか？」ということを明示する情報を指す。
ただし現在の実装ではユーザが見ている全てのページの URL を保存するのではなく、ユーザが最後にアクセスしたページの URL のみを保存している。これは現段階でリクエストラインの有効な解析法を模索中のためである。ログファイルは次のような形式で作成される。

ファイル名　:「(IP アドレス).txt」

内容　:「(URL)」

例えば「IP アドレス：127.0.0.1」のユーザが google のエントランスページを見ていた場合は次のようになる。

ファイル名　:「127.0.0.1.txt」

内容　:「http://www.google.co.jp/」

ログファイルの詳しい用途は 4.3 節で説明するが、チャットサーバでしか利用されないので将来的にはファイルとして保存するのではなくデータとして直接チャットサーバに送ることになる。なお現段階では開発時のデバッグ用データとしてもファイルを利用している。

4.2.4 並列処理

プロキシサーバは複数のユーザが同時に接続する事態が考えられる。シングルスレッドではアクセス数が多くなれば多くなるほどユーザのウェイトタイムも大きくなり、ユーザを不快にさせてしまう恐れがある。この対策としてサーバをマルチスレッド設計にすることでクライアントを並列処理する (図 4.5)。

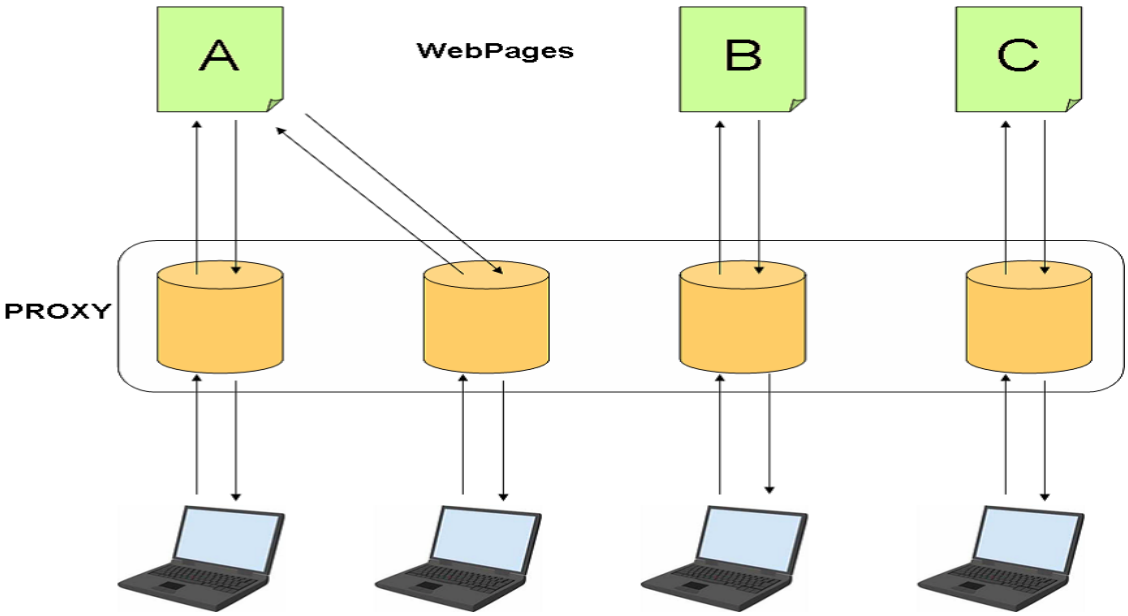


図 4.5: マルチスレッドプロキシサーバ

4.3 実装：チャットサーバ

4.3.1 概要

チャットサーバはソケット (SocketServer クラス) を用いてクライアントアプリケーションと通信を行う。概略図を図 4.6 に示した。チャットサーバは存在する全てのチャットルームオブジェクト (4.3.3 節) と接続中の全ソケットに対するユーザスレッドオブジェクト (4.3.2 節) を保持する。チャットルームは入室中のユーザのユーザスレッドオブジェクトをそれぞれ保持する。ユーザスレッドがソケットのクライアントアプリケーション (一対一対応) との通信を行

うので、入退室・発言・ログアウト等のやり取りはユーザスレッドを通じてチャットルーム・チャットサーバオブジェクトで行われる。
チャットサーバの役割はチャットルームとユーザ全体の管理である。チャットサーバはユーザスレッドを複数保持すること (マルチスレッド) により個々のクライアントを並列処理する一方で、クライアントソケットの接続を待ち続けて新たなソケット接続があれば新規ユーザスレッドを生成する。また必要に応じてチャットルームオブジェクトを生成・破棄することが可能である。さらに個々のチャットルームの入室者数を数えてホットページランキングを作成するメソッドを有している。

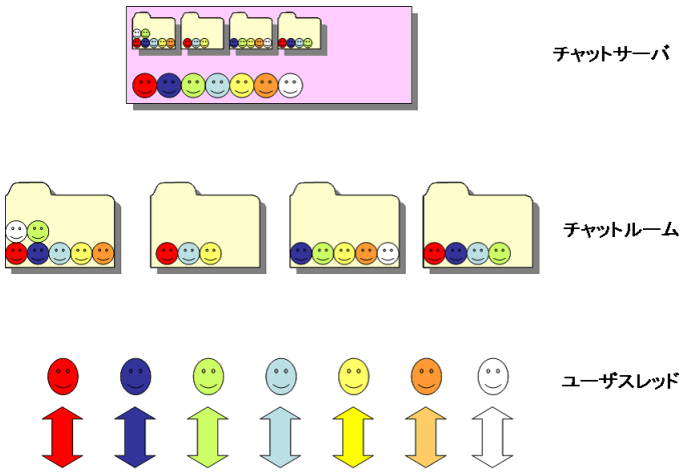


図 4.6: チャットサーバの概略図

4.3.2 ユーザスレッド

ユーザスレッドはクライアントアプリケーションと一対一対応で通信を行っていて、現在クライアント側の会話スペースで表示している部屋の名前 (URL) を保持している。またチャットで使用するユーザの名前を保持し、それを返り値とする外部から呼び出し可能なメソッドを実装している。
またクライアントとの通信はテキストベースで行われ、通信メソッドも外部から呼び出し可能である。通信するメッセージは **[コマンド名 (半角スペース) 内容]** で構成されていて、例えばユーザが「こんにちは」と発言した場合は **[msg こんにちは]** というメッセージを受け取る。ユーザスレッドが受け取る各種コマンドの用途と付随して送られてくる内容の一覧は図 4.7 に示してある。

退室・発言・会話スペースに表示する部屋の変更等のメッセージをクライアント側から受信するとユーザスレッドは対応するチャットルームオブジェクトへアクセスしメッセージの処理を行う。会話スペースに表示する部屋の変更の場合は、新しく表示する部屋で起こった会

話の履歴 (メッセージログ) をクライアント側に送信する。
またユーザスレッドはクライアントソケットの IP アドレスを利用してクライアントのログファイル (4.2.3 節) の更新状況を監視する。ログファイルが更新されるとクライアントが新しい部屋 (URL) に移動したと判断して新しい部屋への入室処理を行う。図 4.8 は入室処理時のクライアント側とのやり取りの説明である。

コマンド名	内容	用途
msg	[メッセージ]	[メッセージ]と発言
setName	[名前]	クライアントの名前を[名前]に変更
view	[部屋名]	[部屋名]のメッセージログを取得
exitRoom	[部屋名]	[部屋名]の部屋から退室
getUsers	[部屋名]	[部屋名]に入室中のメンバーリストを取得
close	[]	ユーザのログアウト

図 4.7: ユーザスレッドが受け取るコマンド一覧

まずクライアントがブラウザで新しいページに移動するとプロキシサーバによってその URL が取得されログファイルに保存される。チャットサーバ内でそのクライアントを監視しているユーザスレッドがログファイルの更新を確認すると、チャットルームの入室メソッド (4.3.3 節で後述) を呼び出し入室メンバーに登録されたあと、チャットクライアントアプリケーションにも入室を通知する。クライアント側は GUI コンポーネントのユーザリストを取得するために、サーバ側に入室した部屋のメンバーを要求する。サーバは要求がくると対応する部屋のメンバーリストをクライアントに送信する。メンバーリストを受け取ったクライアントはそれを元にコンポーネントを更新する。メンバーリストと同じようにクライアント側は部屋のメッセージログを要求・取得する。この時ログファイルの内容 (URL) が既に入室済みの部屋であり二重に同じ部屋の入室メソッドを行う場合があるが、その場合はチャットルームオブジェクトが判断して二重入室を防ぐ。

4.3.3 チャットルーム

チャットルームは部屋名 (URL)・メッセージログ (会話の履歴)・入室中のユーザスレッド配列を保持する。また入退室メソッドを持ちユーザスレッドからの呼び出しが可能である。入室中のユーザスレッドのメソッドを用いることで入室中のメンバー全員の名前をリスト化するメソッドを有していてこれも外部から呼び出し可能である。同様に外部呼出し可能であり

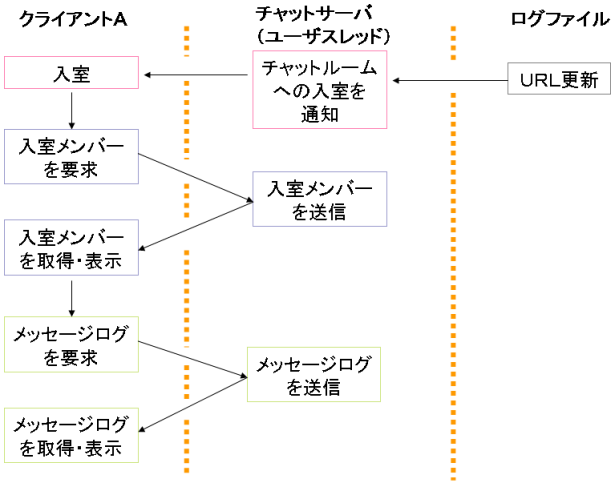


図 4.8: 入室の場合

メッセージログを返り値とするメソッドも有する。入退室メソッドは呼び出されると保持しているユーザスレッド配列において呼び出し元のユーザスレッドの登録・抹消を行う。入室メソッドに関しては 4.3.2 節で述べた様に、呼び出し元のユーザスレッドが既に配列の中に登録されていた場合は二重の登録を防ぐ。またチャットルームオブジェクトはユーザスレッド配列が 0 になるとチャットサーバからの参照を消去して自動的に破棄される。チャットルームオブジェクトとユーザスレッドの関係の例として入室時 (図 4.9) と会話時 (図 4.10) のやり取りを説明する。

入室時

まずユーザスレッド A がクライアントの入室をログファイルの更新から判断する。A は入室メソッドを呼び出すことで自身のオブジェクトをチャットルームに登録する。チャットルームは入室メソッド呼び出しから新たに呼び出し元の A を登録し、ユーザスレッド配列の各要素の通信メソッド (4.3.2 節) を呼び出して、A が入室したことを入室者全員に知らせる。その後図 4.8 でも説明した通り、A は入室メンバーのリストとメッセージログの要求メソッドを呼び出しこれらを取得する。

会話時

まずクライアント側からメッセージを受け取ったユーザスレッドがチャットルームに通知する。その後チャットルームはユーザスレッド配列の各要素の通信メソッド (4.3.2 節) を呼び出して A が発言したことを通知し、その内容をメッセージログに保存する。

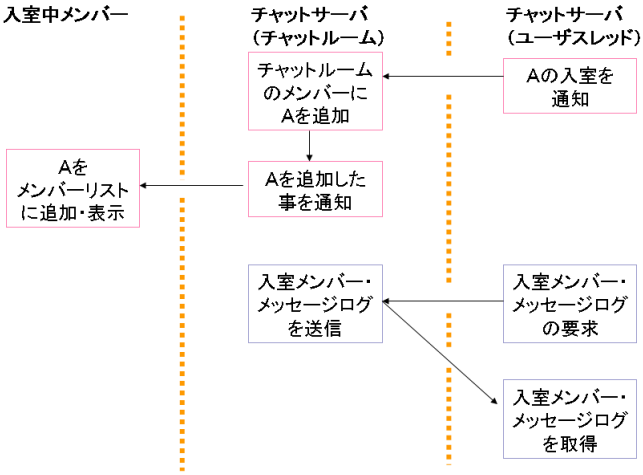


図 4.9: 入室時のやり取り

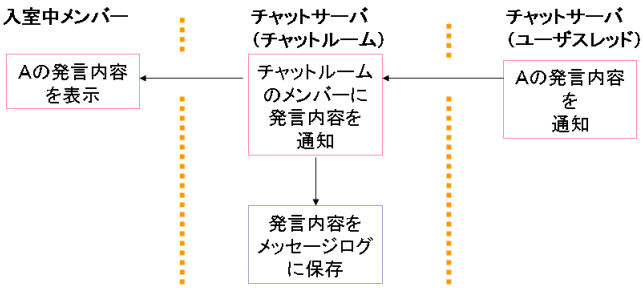


図 4.10: 発言時のやり取り

またチャットルームの登録ユーザ数に変更があった場合には、ホットページランキングを計算するメソッド (4.3 節) を呼び出す。このメソッド自体はチャットサーバ自身が有しているのでチャットサーバが管理しているユーザ全体にランキングの変更を通知することが可能である。図 4.11 はその工程を表している。

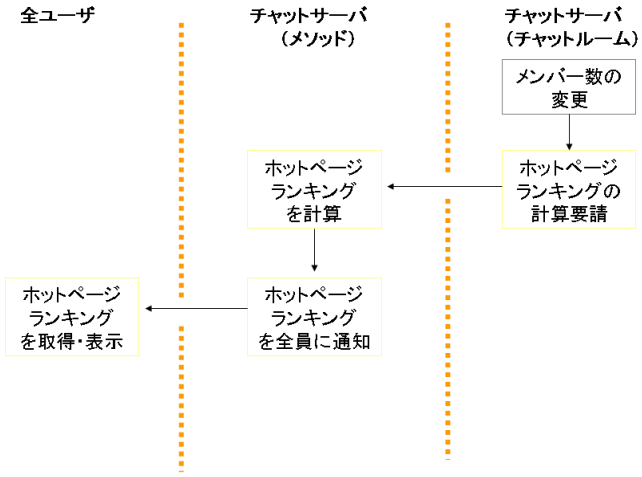


図 4.11: ホットページランキング変更の工程

4.4 実装：クライアントアプリケーション

クライアントアプリケーションはルックアンドフィールの観点からウィンドウコンポーネントには Swing を使用している。またソケット (Socket クラス) を用いてチャットサーバと通信を行うが、その通信は 4.3.2 節で述べたようにチャットサーバ内のユーザスレッドの 1 スレッドとして処理される。ユーザスレッドとの対話でクライアント側が受け取るメッセージのコマンドの一覧は図 4.12 で示されている通りである。コマンドの一部を紹介する。

viewS

このコマンドはクライアント側から view(図 4.12) を送信した場合に、チャットサーバ側からのレスポンスとして受信する。クライアントが view コマンドを送信したということは、クライアントは任意の部屋の会話履歴を会話スペースに表示しようとしていることになる。 この時今まで会話スペースに表示されていた内容を破棄してから新しい会話履歴を表示しなければならない。その最初の工程、つまり会話スペースにある古い内容の破棄を指示するコマンドである。従ってコマンドに付随する内容は無い。

view

このコマンドは viewS に続いて受信する。コマンドに付随する [メッセージ] を会話スペース

のコンポーネントに追加する。

otherRoom
現在会話スペースに表示している部屋以外の場所で会話が起こった場合に受信する。[部屋名] は会話が起こった部屋の名前である。[部屋名] で会話が起こったことを示すためクライアントはルームリストにある対応する部屋の VIEW セルの値を「VIEW」に変更する。

コマンド名	内容	用途
msg	[メッセージ]	[メッセージ]と会話スペースに発言
viewS	[]	会話スペースを初期化
view	[メッセージ]	[メッセージ]と会話スペースに追加
enterRoom	[部屋名]	[部屋名]をルームリストに追加
users	[ＡＢＣ・・・]	入室中のメンバーリストを[ＡＢＣ・・・]と変更
otherRoom	[部屋名]	[部屋名]のVIEWセルの値を「VIEW」にセット
Top1	[部屋名 人数]	[部屋名]がホットページランキング1位で[人数]人いる

図 4.12: クライアント側が受け取るコマンド一覧

Top1
ホットページランキング 1 位の [部屋名] と [人数] を受信する。クライアントは人数に応じて描画する炎のレベルを決定し、コンポーネントの更新を行う。また類似コマンドに [Top2],[Top3] がありそれぞれホットページランキング 2 位,3 位を通知する。4.3.3 節 (図 4.11) で述べたようにチャットルームの人数が変わるたびに通知される。通知は Top1,Top2,Top3 の順に行われる (図 4.13)。

尚ホットページランキングにおける炎の描画に関してはフリーウェアアプレットを開発している四井 賢一郎氏のホームページ[1]に記載されているアルゴリズムを参考にした。

[1]四井 賢一郎氏のホームページ：http://www.kdn.gr.jp/ shii/java/exam/algorithm/

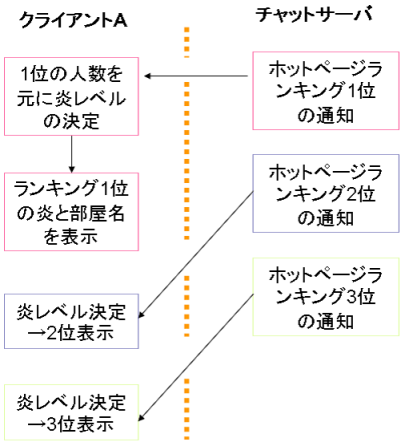


図 4.13: ホットページランキングの受信

第5章　考察と課題

Page-Chat ではインフォーマルコミュニケーションが起きる前段階におけるきっかけとして「偶然同じページを見ているという繋がり」を提示した。しかし Page-Chat はコミュニケーションが始まった後の支援についてはまだまだ課題が残る。例えば会話中のコンテクストアウェアネスに関しては西田 [10] らによって研究されている。西田らの Lock-on-Chat では画像を共有し、それら画像の特定箇所に結び付けられた会話を楽しむことができる。特定箇所にユーザが注目していてその箇所についての会話が起きているという強いコンテクストアウェアネスを受けられる。これは Page-Chat についても応用の余地が考えられる。現在見ている Web ページのどの部分に対して会話が行われているかというアウェアネスを提示することでより充実したコミュニケーションが可能になると予想できる。同様にコミュニケーションにおけるユーザ間のコンテクストアウェアネスの研究は TangibleChat[14],ConChat[8],Notification collage[7],Chat Circles[9] など盛んに行われている。
また会話中及び会話後のコミュニティ作りに関しても課題が残る。Page-Chat では相手と会話をした後にその相手とは特にコミュニティを形成するわけでもなくその場で別れてしまう。気が合う仲間とは何らかのグループ・コミュニティを形成できる機能を拡張していくことでよりソーシャルな要素を取り入れることになり Page-Chat は充実したコミュニケーションツールになると考えられる。さらに形成後のグループ・コミュニティの管理も考慮に入れるべきだろう。これについてはグループの参加メンバーを手軽に変更できる QuickML[11] や、異なるコミュニケーションツールを統合した qwikWeb[12]、インターネット上でも情報の公開範囲を詳細に設定・変更できるようにする enzin[13] などがそれぞれアプローチしていて我々も学ぶべき所が多い。

第6章　関連研究

6.1　繋がりを支援する研究

我々と同じ繋がりに着目した赤塚大典 [5] らの「わくらわ」という研究がある。日常的に付き合いのある知人を「強い紐帯」、普段は疎遠な知人を弱い紐帯に喩えていて、弱い紐帯から何かのきっかけで思いがけずに有益な情報が得られる可能性を述べている。しかし弱い紐帯は普段疎遠なために維持することが難しく従来のコミュニケーションメディアにはこれを支援するツールがまだあまり検討されていない段階である。そこで弱い紐帯を維持しながらも、きっかけが与えられたときにそれを活かすことができる枠組みを目指す試みが行われている。具体的には図 6.1 のように同じページにいる知人の存在をアイコンで提示する。このアイコンは例え気付かなくてもおかしくない位置にある。そのためこのシステムの核心部分は「知人の発見後どのような行動を取るかはユーザ自身に委ねられる」ということである。
「わくらわ」は同じページを見ているという繋がりに着目している点では我々のアプローチと共通している。しかし「わくらわ」は既に本人が属しているコミュニティ内の知人とのきっかけを支援するのに対し、我々は不特定多数の他人とのコミュニケーションを目的としていて積極的な繋がりを支援するという点でコミュニケーションの対象に対するスタンスの相違がある。また「わくらわ」はきっかけを提示するのみでその後のコミュニケーション機能 (メール・チャットなど) は既存のツールに委託している。

図 6.1: わくらわ

6.2　Web ブラウジングに関する研究

井上恭輔 [6] らの **antwave** はグリッドブラウジング (図 6.2) という手法を用いることでブラウザ同士が有益な情報を共有し、他者がどのようなページを選んで遷移していったか？という情報を 3 次元で提示するシステムである。特に目的もなく Web ブラウジングしているユーザにとっては他者が歩んだ道筋は自分をナビゲートする道しるべとなるのである。その際提示されたイメージでは多くのユーザが通った道ほど太いラインで結ばれていて、より人だかりを感じることができる (図 6.3)。
人が集まるページを提示するという点で本研究のホットページランキング機能と共通する部分がある。しかし井上らは人に会うという目的が根幹にあるので、Web ページでの遭遇に対して作為的に会うという立ち位置であるのに対し、我々はそれに加えて偶然ばったり会ってしまうという偶然性に焦点を当てている点で違いがある。

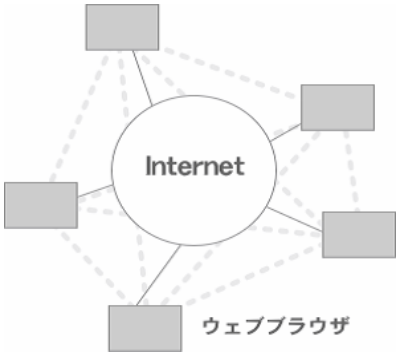


図 6.2: グリッドブラウジング.

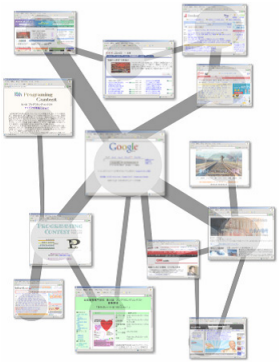


図 6.3: antwave の実行画面

第7章　まとめと今後の展望

Web ブラウジングという日常的な活動の中において「偶然同じページを見ている」という繋がりを構築することで、インフォーマルコミュニケーションのきっかけを支援する Page-Chat のプロトタイプを提案した。Page-Chat は与えられたきっかけをコミュニケーションの誘発に活かすために「部屋を探すことなく会話ができる」チャットシステムを有していて、コミュニケーション生起の前段階における支援をしている。今後は会話中にユーザ間のコンテクストアウェアネスを提示することでより充実したコミュニケーションを行えるようにしたいと考えている。また会話中及び会話終了後における気の合う仲間とのグループ・コミュニティの形成支援にも着手する予定である。

謝辞

本研究の執筆にあたり、筑波大学システム情報工学研究科 田中二郎教授には懇切なご指導とご助言をいただきました。ここに深く感謝致します。また、三末和男助教授、志築文太郎講師には大変有益な議論の機会を与えて頂きました。心から感謝申し上げます。さらに高橋伸講師には、研究内容に留まらず進捗等についてもご指導頂きました。心から感謝致します。筑波大学システム情報工学研究科 インタラクティブプログラミング研究室のメンバーの方々にも大変お世話になりました。この場を借りて厚くお礼申し上げます。

参考文献

[1] 小野孝之, 三村和, 川原圭博, 森川博之, 青山友紀. ”インフォーマルコミュニケーションを支援するプレゼンス技術”. 電子情報通信学会技術研究報告,NS2004-296,March 2005

[2] 北陸先端科学技術大学院大学. ”ナレッジ・サイエンス”(2002).
http://www.kousakusha.com/ks/index.html

[3] B.A.Nardi,S. Whittaker and E. Bradner. “ Interaction and Outeraction: Instant Messaging in Action ”. CSCW’00,pp.79-88,2000

[4] 浜屋敏, 吉田倫子, 土屋太洋. ”ブログ・SNS の双発的特性と組織へのインパクト”. 富士通総研経済研究所『研究レポート』,No.269,June 2006

[5] 赤塚大典. ”弱い紐帯に注目したコミュニケーションメディア「わくらわ」”. WISS 2006,pp.139-140

[6] 井上恭輔. ”antwave-超次元コラボレーションブラウザ”. 第 16 回全国高等専門学校プログラミングコンテスト自由部門,20045,March 2005

[7] Saul Greenberg,Michael Rounding. ”The notification collage: posting information to public and personal displays”. SIGCHI,pp.515-521,2001.

[8] Anand Ranganathan, Roy H Campbell, Arathi Ravi, Anupama Mahajan. ”ConChat: A Context-Aware Chat Program”. IEEE Pervasive Computing,pp.51-57,July 2002.

[9] Fernanda B. Viegas,Judith S. Donath. ”Chat circles”. SIGCHI,pp.9-16,1999.

[10] Takeshi Nishida, Takeo Igarashi. ”Lock-on-Chat: Boosting Anchored Conversation and its Operation at a Technical Conference”. INTERACT 2005 (Short paper), pp.970-973.

[11] Toshiyuki Masui, Satoru Takabayashi. ”Instant Group Communication with QuickML”. Proceedings of the ACM Conference on Supporting Group Work (Group ’03), pp. 268-273, November, 2003.

[12] 江渡浩一郎, 高林哲, 増井俊之. ”qwikWeb: メーリングリストと Wiki を統合したコミュニケーション・システム”. インタラクション 2005, pp. 13-20,February, 2005.

[13] 永田周一, 安村通晃. ”Enzin: 情報の公開範囲を手軽に変更できるコミュニケーションツール”. ソフトウェア科学会 WISS2005, pp.111-116, (Dec. 2005).

[14] 山田裕子, 平野貴幸, 西本一志. ”Tangible Chat: 打鍵振動の伝達によるキーボードチャットにおける対話状況アウェアネス伝達の試み”. 情報処理学会論文誌, Vol.44, No. 5, pp.1392-1403, 2003.