実世界型電子付箋システムにおける漂流型情報共有機構について

伊藤 栄俊　大囿 忠親　新谷 虎松

我々は，物理付箋と電子付箋を用いた発想支援のための実世界型電子付箋システム Mahoca の開発を行っている．本研究では，Mahoca を使用している各ユーザの画面上に，テキストや画像情報を表示する漂流型オブジェクトを用いた漂流型情報共有機構を開発した．漂流型オブジェクトは，各ユーザの画面間を漂うことで，情報の共有とコミュニケーションを支援し，ユーザ間のインタラクションを活発化させることが目的である．深層学習を始めとする機械学習 API を呼び出すためのメッセージ通信を開発した．これにより，機能の拡張性を高める．また本システムは，シングルウィンドウアプリケーションとして開発した．シングルウィンドウアプリケーションは，マルチウィンドウアプリケーションに比べメモリ使用量を抑えることができ，拡張性も高い．本稿では本システムの概要と有効性を示す．

We are developing a reality-based e-sticky notes system “Mahoca” for idea support using physical sticky notes and electronic sticky notes. In this research, we developed a floating mechanism with a floating object that displays text and image information on the screen of each user using Mahoca. The floating object supports information sharing and communication by walking between the screens of each user, and activate interaction among users to lead to idea support. We also developed machine learning API, including deep learning, based on message passing for improving the scalability. Additionally, our system employs a single window application because a single window application uses less memory than a multi window application. Moreover, the single window architecture achieves high extendibility. This paper presents its outline and its effectiveness.

1　はじめに

会議を行う際に，付箋の紙を用いられることがある．アイディアを付箋に書き出し，ホワイトボードなどに貼り付ける．付箋は自由に記述，配置ができ，チーム内でアイディアを共有しやすいという利点がある．過去に行われた会議における，紙の付箋の内容だけでなく，配置方法や付箋のグループといった情報は，現在の会議の参考にするときに有益である．しかし，再利用を考えた場合，電子化された付箋の方が，過去の状態を保存，管理できる点で優れている．我々は実世界型電子付箋システム Mahoca [1] を開発した．Mahoca は，物理付箋を取り込み，電子付箋に変換したり，直接電子付箋を作成したりするシステムである．Mahoca を用いて会議を行う際，過去の会議の情報を用いて関連情報を推薦することは，アイディア出しなどにおいて特に重要である．そこで，我々は，Mahoca における関連情報を提示するためのシステムを開発している [2]．アイディア出しを支援する情報として，過去の会議の情報をもとに，関連する語や画像を推薦することで，アイディアの広がりを支援した．

Mahoca における関連情報の推薦において，以下の 3 点が課題となった．一つ目は，提示された関連情報が作業の妨げにならないことである．例えば，関連情報として画面上にキーワードを提示する場合，提示されたキーワードにより他の情報の視認性が悪くなれば，作業の妨げになる．また，価値が低い情報が画面上に残り続ける状況は避けなければならない．二つ目は，グループ間で情報が共有されることである．ブレインストーミングなどを用いた会議の場合，グループ

A Floating Mechanism for Information Sharing on a Reality-based e-Sticky Notes System

Eishun Ito, 名古屋工業大学工学研究科情報工学専攻, Department of Computer Science, Graduate School of Engineering, Nagoya Institute of Technology.

で作業に取り組むことになる．こうした作業を行っているグループ内において情報共有は重要である．三つ目は，提示される情報に対する機能の拡張性である．テキストだけでなく，画像や音楽など様々な形式の情報を関連情報として提示する場合，それぞれの形式に沿ってユーザに情報が提示できること，また，関連情報を生成するための機構の機能拡張が重要となる．こうした課題に対して，本研究では漂流型情報共有機構を開発した．

本システムでは，漂流型オブジェクトを用いた自由度の高い表現オブジェクトの提供，グループでの送受信が可能な通信システム，拡張性の高い，学習モデルに対するメッセージ通信といった，これらの機能を実装することで課題の解決を行った．これらの機能は，Mahoca だけでなく，様々なシステムの開発を支援することが期待される．

本研究における課題を解決するために，漂流型情報共有機構はシングルウィンドウ方式を採用している．本システムを実装するにあたり，マルチウィンドウアプリケーションとして開発するか，シングルウィンドウアプリケーションとして開発するかが一つの争点となった．システムをシングルウィンドウ方式とマルチウィンドウ方式としてそれぞれ実装し，評価実験を行い，パフォーマンスの評価を行った．結果，シングルウィンドウ方式の方がメモリ使用量を抑えることができ，開発における拡張性も高いため本システムでは，シングルウィンドウ方式を採用した．シングルウィンドウ方式を採用するにあたり用いたアルゴリズムの紹介と，マルチウィンドウ方式と違いを示す．加えて，本システムを用いて作成されたアプリケーションを紹介することで，本システムの有効性を示す．

本稿の構成を述べる．1.1 節では，関連研究の紹介を行い，他の研究との比較により本研究の立ち位置を示す．1.2 節では，実世界型電子付箋システムである Mahoca の紹介を行う．2 章では，開発した漂流型情報共有機構の機能について述べる．3 章では，2 章にて提案したアルゴリズムに基づくパフォーマンスの評価を行ったので，その結果を示す．4 章では，評価実験にもとにした考察，および，開発したシステムに関して応用例をしながら有用性について言及する．5 章にて，本稿を締める．

1.1 関連研究

Mahoca のような実世界とのインタラクションを考慮したシステムの関心は高まっている．Shi らの IdeaWall [3] では，ブレインストーミングを用いた会議中における音声入力情報をリアルタイムにビジュアライズ化するシステムを開発した．IdeaWall では，音声入力を元にテキストや画像による情報推薦を行い，アイディアを広げる支援を図った．会議中の音声入力を元にした関連情報推薦の研究は他にもされており，Andolina らの InspirationWall [4] などが挙げられる．関連情報を提示する上で，IdeaWall では，3 種類の情報提示手法が提案された．IdeaWall は，Mahoca と同じように，ブレインストーミングなどを行っている最中に議論に関連する情報を提示することで，アイディア出しを支援している．しかし，Mahoca とは違い，電子付箋などの情報は画面上に存在しない．そのため，提示方法は限定されないが，例えば，会話の様子を画面上に提示しながら議論を行う状況等では，本研究で取り組んだ問題と同様の問題が発生する．

Martinez-Maldonado らは，様々な対話型のインタフェースを用いたグループ作業に対して，Activity-Centred Analysis and Design(ACAD) フレームワークを利用した分析のための実用可能なアプローチを提案した [5]．Martinez-Maldonado らは，様々な物理的ないしデジタルな機器が利用できる空間におけるグループ作業を想定し，実験を行った．ACAD フレームワークに基づく 4 種類の要素を用いて分析を行うことで，様々なインタフェースを用いたグループ作業を評価している．グループワークにおける評価方法は他にもあり，例えば，Activity-Based Computing(ABC) framework [6] などが挙げられる．

小林らの Chatplexer [7] では，発表中におけるチャットや Twitter などの口頭質問に注目し，重要な発言だけを抽出する試みを行った．Chatplexer における研究で特筆すべき点は，発表における口頭質問と，チャット上におけるテキスト情報を収集し，クロスチャネル返信という概念に基づき，対応関係を明確化し，記録を行っていることである．議論中には口頭に

よる発言が存在している．口頭による議論中有の情報とシステム上の情報との対応関係は，議論の活発度や成果を分析する上で重要な要因である．発表中におけるこうした，発言を発表に活用することは，発表支援につながり有用である．しかし，チャットのテキスト情報などを発表画面上に提示する方法として，スライド画面の最前面や側に関連する情報を提示することが挙げられる．ここで，提示される情報が発表の妨げになってはならない．議論や発表における作業画面の妨げにならない，情報推薦のための提示機構の必要性があげられる．

こうした，会議中における情報を元に関連情報を提示し，議論を活発化させる研究は現在でも多く取り組まれている．スクリーンや大画面ディスプレイを用いることで情報共有を図るケースが多いが，本研究で想定するユースケースは，情報共有するためのスクリーンや大画面ディスプレイの他に，参加者各個人が端末を所持しており，物理的な付箋情報の取り込みや情報の閲覧を行う状況を想定している．こうした状況である場合，共有画面だけでなく，各個人が使用している端末ごとの作業画面に対する情報推薦が重要となる．

1.2　実世界型電子付箋システム Mahoca

実世界型電子付箋システム Mahoca は，アイディアの再利用のために電子付箋にアイディアを入力し，会議参加者間共有するためのシステムである．Mahoca では，会議参加者が「スクリーン」と「クライアント」の二つのアプリケーションを利用することで，電子付箋を利用した会議を行う．図 1 に Mahoca を使用している様子を示す．スクリーンは，公開された電子付箋を画面に表示し，会議参加者に共有するアプリケーションである．スクリーンでは，会議に対応する「ベースボード」と呼ばれる画面を全面に表示する．ベースボード上には会議参加者が公開した電子付箋が表示される．

クライアントは，電子付箋を作成，または編集してベースボードに公開するために利用されるアプリケーションである．クライアントは，各会議参加者が持つスマートフォンやタブレット端末上で動作する．電子付箋には，付箋に入力したテキストや手書きメモの他に，メタ情報としてテキストのメモと任意の URL を入力して付加できる．

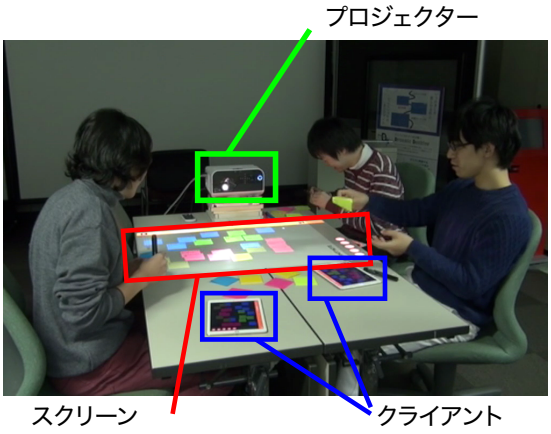

図 1　Mahoca を使用している様子

Mahoca では，会議参加者は，クライアントを利用して自分の意見を書いた電子付箋を作成する．作成した電子付箋をスクリーンで表示したベースボード上に公開することで，他の会議参加者に意見を共有する．ベースボード上の電子付箋で他者の意見を確認し，議論をすることで会議を進行できる．

ここで，Mahoca における課題について述べる．1 章にて Mahoca における課題を三つ挙げた．一つ目は，作業の妨げにならないことである．例えば，関連情報として画面上にキーワードを提示する場合，提示されたキーワードにより他の情報の視認性が悪くなれば，作業の妨げになる．また，価値が低い情報が画面上に残り続ける状況は避けなければならない．この課題を解決するために，我々は，透明ウィンドウを用いた漂流型オブジェクトによる情報提示によりこの問題の解決を図った．

二つ目は，グループ間で情報が共有されることである．ブレインストーミングなどを用いた会議の場合，グループで作業に取り組むことになる．作業を行っているグループ内での情報共有は重要である．スクリーンや大画面ディスプレイに表示されるような情報共有だけでなく，参加者間の端末ごとの作業画面上において，関連情報やチャットのようなメッセージを共有することで，議論の活発化が期待される．こうした，

参加者間のコミュニケーションや情報共有を支援するための通信システムを設計することで，問題の解決を図った．

三つ目は，提示される情報に対する拡張性の向上である．テキストだけでなく，画像や音声など様々な形式の情報を関連情報として提示する場合，それぞれの形式に沿ってユーザに情報が提示できる機構が必要である．そこで，Web ページのレイアウトで利用されている HTML と CSS を用いることで表現の幅を確保した．また，インタフェースだけでなく，関連情報を生成するための機構のスケーラビリティの向上も重要である．Mahoca における関連情報推薦では，Web ニュースコーパスや Wikipedia コーパスを元にした学習モデルを使用している．こうした学習モデルをより簡単に拡張するための仕組みと，ユーザが暗黙的に利用できるサービス形態を考案した．

2 漂流型情報共有機構

章にて提示した課題を解決するために，漂流型情報共有機構を開発した．漂流型情報共有機構には三つの特徴がある．一つ目は，ユーザに対してディスプレイの最前面に表示される漂流型オブジェクトである．二つ目は，一対一やグループでの通信を可能にする通信システムである．三つ目は，深層学習を始めとする機械学習 API を呼び出すためのメッセージ通信である．それぞれの特徴に関して述べる．

2.1 漂流型オブジェクトと透明ウィンドウ

漂流型情報共有機構では，漂流型オブジェクトという画面上を漂うように動くものを用いて情報を提示する．画面上を漂うように動くことで，背後の情報が隠れてしまう状況を防ぐことができる．必要な情報を持つ漂流型オブジェクトをユーザが選択し，選択されなかったオブジェクトは，一定時間毎に画面上から消える．漂流型オブジェクトは，漂流型情報共有機構における通信モジュールを用いて，グループ間で共有される．そのため，画面上だけでなく，通信システム上において，メッセージのやり取りによりユーザ間で情報が伝達され，モバイルエージェントのような働きをする．

漂流型情報共有機構は Electron[†1] と呼ばれるフレームワークを用いて開発を行った．Electron は，デスクトップアプリケーションに Web アプリケーションの開発手法を応用すること可能である．そのため，漂流型オブジェクトの形状は，HTML と CSS で定義できる．これにより，漂流型オブジェクトで表示できる媒体として，テキスト情報だけでなく，画像や URL，音声や映像も表示することができる．HTML や CSS は Web サイトのデザインなどに用いられており，漂流型情報共有機構は Web アプリケーションの提供も可能である．

漂流型オブジェクトは，透明ウィンドウ上に表示される．透明ウィンドウとは，デスクトップ上に表示される透明なウィンドウである．ウィンドウレベルが調節されており，デスクトップ上において最前面になるよう配置される．そのため，他のウィンドウよりも優先されてコンテンツが画面上に表示される．漂流型オブジェクトを透明ウィンドウ上に割り当てることで，常に最前面でユーザに情報を提示することが可能となる．

漂流型オブジェクトを用いてのシステムの実装を行う際に，オブジェクトの割り当て方として 2 種類の方法を検討した．一つは，漂流型オブジェクト一つに対して，ウィンドウを一つ割り当て，デスクトップ上に複数のウィンドウが生成されるマルチウィンドウ方式である．もう一方は，全ての漂流型オブジェクトをウィンドウを一つ割り当て，デスクトップ上には単一のウィンドウのみ存在するシングルウィンドウ方式である．図 2 に，シングルウィンドウ方式とマルチウィンドウ方式のイメージ図を示す．図 2 中の外側の矩形で示されるのがユーザの作業領域を表し，雲のデザインが漂流型オブジェクトを表し，矩形に囲まれた斜線部分が透明ウィンドウである．右下に示されているのがマルチウィンドウ方式である．左下で示されているのが，シングルウィンドウ方式である．

図 2 の右下に示されているのがマルチウィンドウ方式である．マルチウィンドウ方式では，漂流型オブジェクト毎に透明なウィンドウを生成し，情報を提示

†1 https://electron.atom.io/

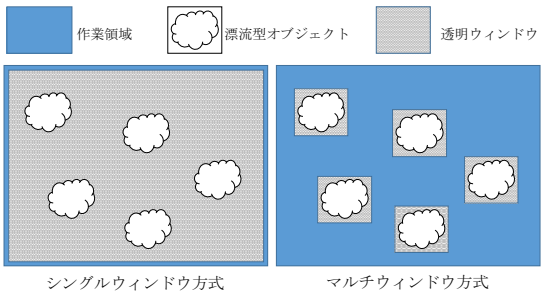

図 2　シングルウィンドウ方式とマルチウィンドウ方式

する．マルチウィンドウ方式の利点は，ユーザとウィンドウ間でのインタラクションの管理が容易で．ユーザが注目したオブジェクトを一意に見つけられる点である．複数のオブジェクトを同一のウィンドウに割り当てる場合，各オブジェクトを区別するための ID が必要となる．マルチウィンドウ方式の場合は，ウィンドウ毎に区別されるため，内部でオブジェクトを区別する機構が不要となり，ウィンドウ内には単一のオブジェクトしか存在しないため，HTML や CSS などの設定が単純化できる．また，ウィンドウサイズは，表示するオブジェクトの大きさ程度で十分となる．

図 2 の左下で示されているのが，シングルウィンドウ方式である．シングルウィンドウ方式では，作業領域全体に透明ウィンドウを生成し，その上に，漂流型オブジェクトをそれぞれ割り当てている．シングルウィンドウ方式の利点は，単一のウィンドウの生成のみを行い，そのウィンドウ上でオブジェクトを生成するため，メモリ使用量がマルチウィンドウ方式に比べ抑えられると期待されることである．また，シングルウィンドウ方式の場合，ただし，オブジェクトの配置範囲に合わせてウィンドウの大きさを決めなければならない．

本システムでは，シングルウィンドウ方式を採用した．理由として 2 点が挙げられる．1 点目の理由は，シングルウィンドウ方式の方が，生成するウィンドウ数を抑えられるためマルチウィンドウ方式に比べメモリ使用量が抑えられると推測されるためである．生成されたウィンドウの面積の合計を比較すると，マルチウィンドウ方式の方は漂流型オブジェクトの数が増えるほどウィンドウの占める総面積の値が大きくなるため，データサイズが膨らみメモリ使用量が増えると考えられる．また，シングルウィンドウ方式では，最初のウィンドウ生成時以降，ウィンドウを新たに生成する必要がなく，漂流型オブジェクトのデータサイズはウィンドウに比べ小さいためメモリ使用量が抑えられると沿い即される．

2 点目の理由は，マルチウィンドウ方式の場合，CSS アニメーションなどの表現に制約がかかるためである．マルチウィンドウ方式の場合，漂流型オブジェクトの位置情報は各ウィンドウの位置に依存する．つまり，オブジェクトの並行移動に関しては，ウィンドウの位置情報を制御する必要があり，平行移動のアニメーション制御するための処理が追加で必要になる．シングルウィンドウ方式の場合，ウィンドウサイズを作業領域全体とすることで，オブジェクトの位置情報を制御することで，領域内であればどこでも配置することができる．また，オブジェクトのアニメーションの設定が CSS アニメーションで制御できるため，様々な既存の CSS アニメーションライブラリを利用することができる．本システムにおいても，TweenMax.js[†2] という CSS アニメーションのライブラリを利用している．

シングルウィンドウ方式の場合，透明ウィンドウを作業領域の大きさで，常に最前面に存在させることになる．つまり，ユーザから行われるマウスやキーボードによる入力が他のウィンドウよりも優先されて伝えられることになる．透明ウィンドウが表示されている間は他のウィンドウの操作の妨げとなるため，漂流型情報共有機構で生成されるウィンドウはユーザに情報を表示する機能のみとし，ユーザからの入力は背面に存在する他のウィンドウに透過するようにした．これにより，作業を妨げずに情報を提示することが可能となる．しかし，イベント透過状態では，ユーザは透明ウィンドウ上に表示されている任意の漂流型オブジェクトとのインタラクションを行うことができない．そこで，ユーザからのイベント情報を監視し，透明ウィンドウに伝えるためのアルゴリズム提案する．

シングルウィンドウ方式における，ユーザからのイ

†2　https://greensock.com/tweenmax

Algorithm 1 シングルウィンドウ方式

```
e ← observeEvent()
p ← getPoint(e)
while o ∈ Objects do
    Hello
    if isCollision(o, p) then
        show(window)
        dispatch(window, e)
    else
        hide(window)
        propagation()
    end if
end while
```

ベント情報を監視し，透明ウィンドウに伝えるためのアルゴリズムを，Algorithm 1 に示す．ユーザからの入力イベントを監視し，イベントの発生座標が透明ウィンドウ上のオブジェクト上であった場合，透明ウィンドウにイベント情報を伝達する処理である．まず，$observeEvent()$ を用いてイベント情報 e を監視する．得られたイベント情報 e から座標情報 p を取得する．座標情報を取得する際は，イベントがディスプレイの作業領域や位置情報が得られるため，透明ウィンドウの座標系に合うよう調整する．取得した座標情報 p を元に，透明ウィンドウ $window$ 上に存在する漂流型オブジェクト群 $Objects$ の各オブジェクト o と衝突判定を行う．イベントが発生した座標 p が漂流型オブジェクト o と衝突していた場合，$show()$ 透明ウィンドウのイベント透過状態を解除し，イベントが伝播されるようにし，$dispatch()$ でイベント情報を透明ウィンドウに伝える．衝突していない場合，透明ウィンドウにイベントが伝播しないよう $hide()$ でイベント透過状態とし，背面に存在するウィンドウに $propagation()$ でイベントを伝播する．

提案したアルゴリズム Algorithm 1 に基づき，ユーザと漂流型情報共有機構を通じて透明ウィンドウと行われるインタラクションにおけるイベント処理について，図 3 を用いて説明する．漂流型情報共有機構には，漂流型オブジェクトの生成とイベント処理の流れに関連するモジュールとして，ユーザの入力イベント

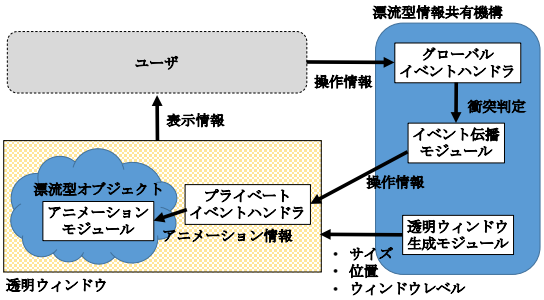

図 3　シングルウィンドウ方式における漂流型オブジェクトとイベント処理

を監視するグローバルイベントハンドラ，入力イベント情報中の座標情報を元に，漂流型オブジェクトと衝突判定を行い，衝突時は透明ウィンドウにイベント情報を伝えるイベント伝播モジュール，ディスプレイの作業領域全体の最前面に透明なウィンドウを生成する透明ウィンドウ生成モジュールが存在する．

漂流型オブジェクトは透明ウィンドウ内で生成される．漂流型情報共有機構から送られるイベント情報をもとに，対応したアニメーションを選択するプライベートイベントハンドラを持ち，これは透明ウィンドウ毎に設定される．アニメーションの例として，画面の端を集会する動きや，フリーハンドで描いた軌跡を追跡する運動が可能である．漂流型オブジェクトが行うアニメーションは透明ウィンドウ毎に設定されるため，使用するユーザごとに動きを変更することが可能で，ウィンドウの配置状況やディスプレイの環境に応じて調整することが可能である．

透明ウィンドウおよび漂流型オブジェクトからユーザに対して情報提示が行われるが，ユーザから透明ウィンドウに対して直接イベントを伝えることがない．そのため，漂流型情報共有機構におけるグローバルイベントハンドラにおいて，ユーザからの透明ウィンドウに対する処理は行われないと定義することで，一方向の情報伝達機構として使用することも可能である．

2.2 通信システム

通信には，WebSocket[†3] を用いている．WebSocket はサーバ・クライアント間で双方向通信を可能にする通信規約の一つである．WebSocket を用いることで高速な情報伝達が可能である．また，クライアント間を通信管理サーバが取り持つことで，クライアントはレスポンスを必用としない一方向な通信も可能である．本システムにおける通信システムでは，クライアントはあらかじめ登録されている，通信管理サーバに対して，識別するための id と，サイト (Site) とトークン (Token) を設定する．Site は，通信を行うクライアントを区別するためのもので，Token に応じてチャンネルが生成される．登録されている正しい Site と Token の値でリクエストを要求するユーザのみが通信を行える．

クライアント間で通信を行う場合，受信側のクライアントに送られてくるメッセージには 3 種類が存在する．

- 特定の相手に送信する，ダイレクトメッセージ (DMSG)
- 所属するグループのメンバー全員に送信する，グループメッセージ (GMSG)
- 同じサイトにアクセスしているメンバー全員に送信するブロードキャストメッセージ (BMSG)

DMSG は，id を指定したクライアントに対して通信を行う際に用いられるメッセージで，一対一通信を行う．GMSG は，同じ Site と Token を持つユーザ内でさらにグループを作りたい場合に使用する．あらかじめデータベースに，所属するメンバー等のグループ情報を登録しておくことで，一対多の通信を実現する．BMSG は，同じ Site と Token を持つ全てのユーザに対してメッセージを送る場合に使用する．

本システムにおける通信のシステム構成図を図 4 に示す．通信を行う各クライアントは漂流型情報共有機構の通信モジュールを用いて通信を行う．通信モジュールでは，クライアント同士を区別するための *id* の設定と，通信管理サーバの *URL* と *Site*，*Token* を用いて通信の確立，送受信を行う．通信サーバから送

†3 https://tools.ietf.org/html/rfc6455

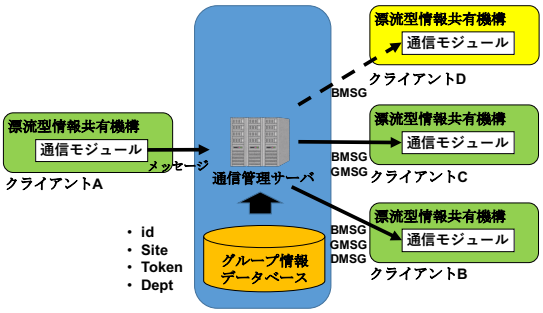

図 4 通信システム

られるメッセージは前述の DMSG，GMSG，BMSG の 3 種類がある．DMSG の場合は，通信相手の id を知る必要がある．GMSG は，事前に登録されている所属グループの値 *Dept* を元にメッセージを送る．BMSG は図 4 の場合，クライアント A はクライアント B の *id* を知っている．クライアント A とクライアント B とクライアント C は同一のグループ *Dept* に所属している．クライアント D は同一の *Site*，*Token* に接続している．そのため，クライアント A はクライアント B に対しては DMSG，GMSG，BMSG が送信でき，クライアント C に対しては GMSG，BMSG，クライアント D に対しては BMSG のみが送信できる．本システムにおけるグループ通信の特徴として，漂流型オブジェクトという，重畳表示が可能で，動的なオブジェクトを用いているため，複数グループからのメッセージを同時に表示することが可能である．そのため，例えばクライアント B はクライアント A から送られていくる DMSG，GMSG，BMSG を全てを同時に画面上に出すことが可能である．

2.3 メッセージ通信

関連情報を生成するために学習済みのモデルを利用することがある．特に，過去のニュース記事など，学習元のコーパスが大きく，関連度の計算に時間が掛かるものや，高い処理能力を求める学習モデルが存在する．こうしたモデルを利用する場合，高速な処理が可能なサーバを用意し，サーバ側で処理を行うことで高速化を図ることができる．

漂流型情報共有機構では，こうした学習済みモデル

を含むサーバ群を管理し，利用者にサーバ群を意識させず，API を呼び出すためのメッセージ通信を提供する．メッセージ通信では，利用したい状況と，利用したい状況に応じたデータを JSON 形式のメッセージを送ることで期待する処理を実行する．これにより，新たな学習モデルを導入する際は，漂流型情報共有機構を管理するサーバに対して，サーバ情報を追加するだけで機能を拡張することが可能となる．

メッセージ通信を用いて利用できる機能として，自然言語処理であれば，MeCab を用いた形態素解析，関連語を推薦するために生成した TF-IDF モデルや Word2vec モデルが利用できる．他にも，Mahoca 上に投稿された画像に対して，どんな画像であるのか判別し，タグを付与する ResNet50 [8] モデルや Incerption-v3 [9] モデルなどが利用できる．

3 評価

2.1 節において，本システムの実装において，シングルウィンドウ方式とマルチウィンドウ方式の 2 種類が存在することを示唆した．双方の性能を比較するために評価実験を行った．評価実験として，シングルウィンドウ方式として実装した際のメモリー使用量と，マルチウィンドウ方式として実装した際のメモリー使用量の計測を行った．シングルウィンドウ方式で生成されるウィンドウは，作業領域と同じ大きさとなるようにウィンドウサイズを設定する．マルチウィンドウ方式で生成されるウィンドウは，表示する漂流型のオブジェクトの大きさに応じたウィンドウサイズとする．実行環境は，macOS Sierra Version10.12.6，2.7GHz Intel Core i5，8GB 1867MHz DDR3，Node.js Version 6.9.5，Electron Version1.6.8 である．

図 5 に評価実験の結果を示す．横軸は，表示している漂流型オブジェクトの数を表している．縦軸は，メモリ使用量 (private bytes) を表しており，単位は kb である．図 5 より，10 オブジェクト以上の時，シングルウィンドウ方式の方がメモリ使用量が少ないことが分かる．

3.1 Mahoca における漂流型オブジェクトの実行例

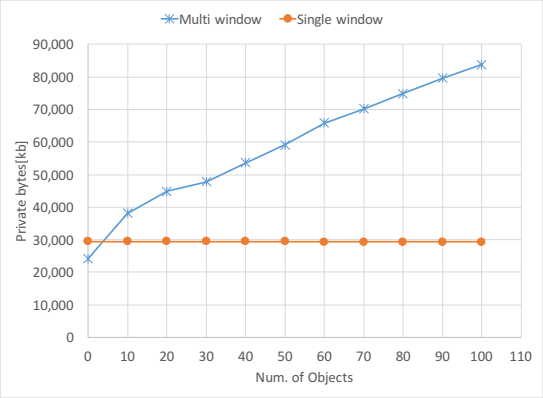

図 5 シングルウィンドウによる実装とマルチウィンドウによる実装に伴うメモリ使用量

図 6 は Mahoca 上における漂流型オブジェクトの実行例を示している．Mahoca を用いて生成された電子付箋情報を元に，関連情報を選択し，漂流型オブジェクトを用いて Mahoca 上に表示している．図 6 中の漂流型オブジェクトには赤色の雲と黄色の雲で表現された 2 種類が存在する．これは，図 6 中の付箋の色情報を元に推薦を行っているためである．漂流型オブジェクトの情報を元に，Mahoca 上の電子付箋を生成することが可能である．

4 考察と応用

本システムにおける貢献として，漂流型オブジェクトを用いた自由度の高い表現オブジェクトの提供，グループでの送受信が可能な通信システム，拡張性のある学習モデルに対するメッセージ通信により，Mahoca だけでなく様々な応用が可能であることが挙げられる．その応用例として上記の通りの例を挙げた．また，評価実験の結果より，シングルウィンドウ方式の方がマルチウィンドウ方式に比べメモリ使用量を抑えられることが示された．他の利点も考慮した結果，本システムではシングルウィンドウ方式を採用した．

透明ウィンドウはレイヤー化することで，デスクトップ上における表現の幅の拡大が期待される．例えば，作業領域の上に，通知やエフェクトなどの読み込

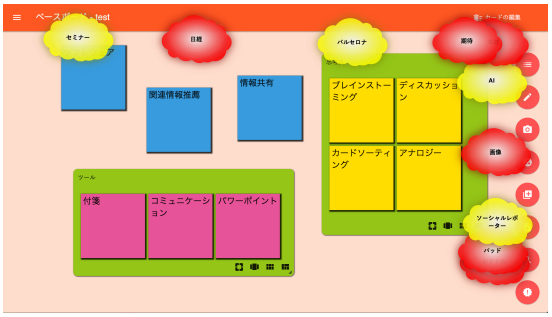

図 6　Mahoca 上における漂流型オブジェクトの実行例: 赤色の雲と黄色の雲で表現された 2 種類の漂流型オブジェクトが最前面に表示されている．漂流型オブジェクトの背面に表示されているのは Mahoca のベースボード画面．

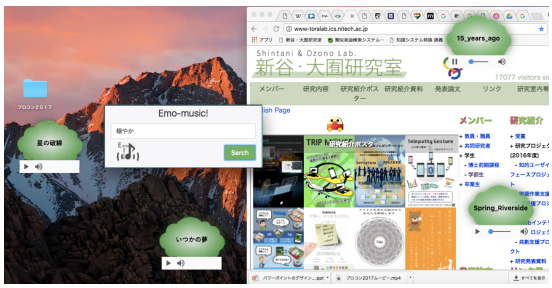

図 7　漂流型情報共有機構の用いた楽曲推薦

み専用のオブジェクトをもつ透明ウィンドウを配置させる．さらにその上に，ポインタなどのユーザとのインタラクションを反映させるオブジェクトを持つ透明ウィンドウを配置することで，ユーザは最上面のオブジェクトに対しては操作可能であるが，2 層目のオブジェクトに対しては権限がなく，閲覧のみとすることができる．このように，透明ウィンドウを重ねて上で，本機構における漂流型オブジェクトのような表示オブジェクトを配置することで，デスクトップ上におけるデザインの幅を広げることができる．

漂流型情報共有機構は Mahoca だけでなく様々な応用が可能である．本システムは，Mahoca と疎結合で構成される機構であるため，他のシステムへの導入が容易である．本システムを有用性を示すために，20 代成人男性 16 名に本システムを用いて様々なアプリケーションを作成してもらった．応用例を 2 つ挙げる．

一つ目は，回覧板システムである．グループに対して通信を行う場合に，グループのメンバー全員に送信するのではなく，グループ内で順序関係を決め，メッセージを漸次送信するシステムである．例えば，学生が執筆した論文の添削では，自分の他に他人からチェックを受けることがある．その際，担当教員に見せる前に，他の同級生や上級生に見てもらうことがある．このように段階的に添削を行う場合，回覧板システムのような順序関係を持つメッセージ送信は有効である．これは漂流型情報共有機構における通信システムの応用で実現される．

二つ目は，楽曲推薦を行うシステムである．本システムで提供している漂流型オブジェクトが用いられている．図 7 に実行例を示す．図 7 では漂流型オブジェクトには音声データが付与されており，再生バーを押すことで，ムードに応じた曲が再生される．これは，漂流型情報共有機構の漂流型オブジェクトやメッセージ通信により支援されている．

他含めの 16 名が，本システムを応用して開発したアプリケーションを総括する．本システムにおける漂流型オブジェクトと通信システムを使うことでチャット機能は容易に実装ができる．加えて，CSS アニメーションによりユーザの興味を惹きつけ，通知機能や関連情報の推薦機能を実装しているものが多かった．こうしたインタフェース部分で特に開発者への支援が行えた．また，ユーザとしては，画面上を漂う物体は視覚的効果が大きく，テキストなどの情報だけでなく，動物やキャラクターなどを用いたアプリケーションの方が，ユーザからの評価が高い結果となった．

5　おわりに

本稿では，Mahoca における課題を解決するために漂流型情報共有機構を開発した．漂流型情報共有機構には三つの特徴がある．一つ目は，ユーザに対してディスプレイの最前面に表示される漂流型オブジェクトである．二つ目は，一対一やグループでの通信を可能にする通信システムである．三つ目は，深層学習を始めとする機械学習 API を呼び出すためのメッセージ通信である．本システムは，漂流型オブジェクトを

用いた自由度の高い表現オブジェクトの提供，グループでの送受信が可能な通信システム，拡張性の高い，学習モデルに対するメッセージ通信により，Mahoca だけでなく様々な応用が可能である．実装において，パフォーマンスの向上のためにウィンドウ管理に関して評価実験を行い，シングルウィンドウ方式の方がメモリ使用量が抑えられたことから，本システムではシングルウィンドウ方式を採用した．また，本システムの有効性を示すために，20 代成人男性 16 名に本システムを用いて様々なアプリケーションを作成してもらったところ，チャット機能や情報推薦において開発者を支援し，ユーザからの関心を惹くシステムであることが示された．

謝辞 本研究の一部は JSPS 科研費 JP15K00422, JP16K00420 の助成を受けたものです．

参 考 文 献

[1] 大囿忠親, 丹羽佑輔, 藤江翔太郎, 渡邊正人, 鈴木智也, 伊藤栄俊, 岩田知, 新谷虎松. Mahoca: 議論事例の再利用を可能にする実世界型電子付箋システム. 電子情報通信学会 人工知能と知識処理研究会, 第 116 巻, pp. 33–38, 2016.

[2] 伊藤栄俊, 鈴木智也, 丹羽佑輔, 大囿忠親, 新谷虎松. ブレインストーミング事例の再利用のための関連情報推薦機構の試作. 2017 年度人工知能学会全国大会（第 31 回）講演論文集, 第 31 巻, pp. 1–4, 2017.

[3] Shi, Yang, Wang, Yang, Qi, Ye, Chen, John, Xu, Xiaoyao, Ma, Kwan-Liu. Ideawall: Improving creative collaboration through combinatorial visual stimuli. In *Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing*, CSCW '17, pp. 594–603, New York, NY, USA, 2017.

[4] Andolina, Salvatore, Klouche, Khalil, Cabral, Diogo, Ruotsalo, Tuukka, Jacucci, Giulio. Inspirationwall: Supporting idea generation through automatic information exploration. In *Proceedings of the 2015 ACM SIGCHI Conference on Creativity and Cognition*, Proceedings of the ACM SIGCHI Conference on Creativity and Cognition, pp. 103–106, New York, NY, USA, 2015.

[5] Martinez-Maldonado, Roberto, Goodyear, Peter, Kay, Judy, Thompson, Kate, Carvalho, Lucila. An actionable approach to understand group experience in complex, multi-surface spaces. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, CHI '16, pp. 2062–2074, New York, NY, USA, 2016.

[6] Bardram, E. Activity-based computing: Support for mobility and collaboration in ubiquitous computing. *Personal Ubiquitous Comput.*, Vol. 9, No. 5, pp. 312–322, September 2005.

[7] 智也小林, 一志西本. Chatplexer：チャットを併用する口頭発表における発表者のための重要発言選択支援の試み. 情報処理学会論文誌, Vol. 53, No. 1, pp. 12–21, jan 2012.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, Vol. abs/1512.03385, , 2015.

[9] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. *CoRR*, Vol. abs/1512.00567, , 2015.