社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE.
OFS2001-13, AI2001-18 (2001-07)

Shine: ネットワークコミュニティ支援システムのエージェント指向フレームワーク

吉田 仙 †　亀井 剛次 †　大黒 毅 ‡　桑原 和宏 †
† 日本電信電話 (株)　‡ (株) エヌ・ティ・ティ エムイー

京都府相楽郡精華町光台 2–4 NTT コミュニケーション科学基礎研究所
phone: 0774–93–5235　e-mail: yoshida@cslab.kecl.ntt.co.jp

あらまし　ネットワークコミュニティを支援するシステムを対象とするフレームワーク Shine について報告する．Shine は，ネットワークコミュニティを支援する各種システムに統一的な枠組みを与え，それらのシステム間での部品の共有や動作の連携を可能にする．また， Shine はエージェント指向のアーキテクチャを持っており，これによってコミュニケーション範囲の分散協調管理機構が実現される．Shine のこれらの性質について述べるとともに，幾つかのシステムへの Shine の適用例を示しフレームワークとしての有効性を確かめる．

キーワード　ソーシャルウェア，マルチエージェントシステム，フレームワーク

Shine: An Agent-oriented Framework of Network Community Support Systems

Sen Yoshida†　Koji Kamei†　Takeshi Ohguro‡　Kazuhiro Kuwabara†
† Nippon Telegraph and Telephone Corporation　‡ NTT-ME Corporation

NTT Communication Science Labs., 2–4, Hikaridai, Seika, Soraku, Kyoto, Japan
phone: +81–774–93–5235　e-mail: yoshida@cslab.kecl.ntt.co.jp

Abstract　A report is presented on Shine, a framework of systems that support network communities. Shine provides a unified structural design approach to various network community support systems and enables them to share software components and cooperate with each other. Furthermore, Shine has an agent-oriented architecure, which achives the distributed cooperative control of communication spheres. These features of Shine are described and its effectiveness as a framework is verified by applying it to several systems.

key words　socialware, multi-agent system, framework

1 はじめに

ネットワークコミュニティを支援するシステムの開発が数多く進められている [2, 7]．これらのシステムはソーシャルウェアと呼ばれる [1]．

著者らは，ソーシャルウェアの様々なアプリケーションシステムの開発において，それらに統一的な枠組みを与え，システム間での部品の共有や動作の連携を可能にするため，ソーシャルウェアのフレームワーク Shine の構築を進めている [9, 8]．このフレームワークの特徴は，エージェント指向のアーキテクチャを持つことによりコミュニケーション範囲の分散協調管理機構が実現される事である．

本報告では，第 2 章においてソーシャルウェアのエージェント指向プラットフォームについて概説し，第 3 章で Shine の具体的な構成を説明する．第 4 章では幾つかのアプリケーションシステムへの Shine フレームワークの適用例について述べる．第 5章でまとめとする．

2 ソーシャルウェアのエージェント指向フレームワーク

ソーシャルウェアとは，ネットワークコミュニティを支援するコミュニケーションシステムの総称であり，そのアプリケーションは色々な種類のものが考えられる．例えば，推薦システムや協調フィルタリングのような人から人への口コミに基づく情報流通，ブレインストーミングなどのインフォーマルで創造的な会話，効果的な出会いの提供や潜在的なコミュニティに気付かせること，親密さやコミュニティ感覚の維持などを支援の対象とするアプリケーションシステムがある．

このようにソーシャルウェアには様々なアプリケーションが考えられるが，これらのアプリケーションの間には幾つか共通する機能が存在する．すなわち，コミュニティにおける各人の情報をうまく格納し利用する機能や，メンバが流動的なコミュニティに適応する柔軟なコミュニケーション手段といったものである．しかし，これら共通の機能をまとめた統一的な枠組みは，現在のところ存在しない．

ところで，近年，ソフトウェア開発におけるソフトウェアアーキテクチャの重要性が認識されるようになってきている [5]．ソフトウェアアーキテクチャには以下のような利点があると言われる．すなわち，1) 設計時に統一的な枠組みを与え，2) 部品の共有や再利用，あるいはシステムの統合や協調を可能にし，3) 開発者間の意思疎通の助けになる．ソーシャルウェアのアプリケーションシステムにおいても，それに適したソフトウェアアーキテクチャを提供できるフレームワークが存在すれば，上記のような利益を享受でき，より豊かなソーシャルウェア環境の実現が期待できる．

このような動機から，我々は，様々なソーシャルウェアアプリケーションの共通の基盤として， Shine というフレームワークを開発している．

Shine の特徴は，エージェント指向のアーキテクチャを持つことである．本章の以下の部分で，エージェント指向アーキテクチャがソーシャルウェアに有効な理由と，我々が取った Shine のアーキテクチャのエージェント化の方針について説明する．

ネットワークコミュニティは，実世界のコミュニティと比べて，地理的あるいは時間的制約を受けずにコミュニケーションができるという強みがある．しかしながら，この制約からの開放は，しばしば情報過多やコミュニティの肥大化をもたらす．それを防ぐには，コミュニティに参加する人を適切に選別しコミュニケーションの範囲を適度に保つことが求められる．ソーシャルウェアでは，こうしたコミュニケーション範囲の管理に如何に取り組むかが重要である．

実社会では，人々は一対一の人間関係をベースとし，仲介や自己紹介を通じて知らない他人と会う．その結果，人々は知人関係を柔軟に変更し，各人に見合ったコミュニティを他人と協調しながら構成していく．実社会におけるコミュニケーション範囲の管理はこのように分散的かつ協調的に行われる．このようなコミュニケーション範囲の分散協調管理機構は，ネットワークコミュニティに持ち込んでも効果的に機能するであろう．このことから我々はコミュニケーション範囲の分散協調管理機構の実現を目指す．

ところで，これまでに開発されているソーシャルウェアアプリケーションシステムの全体的なアーキテクチャの設計方針に目を向けると，ほとんどのシステムではクライアントサーバ型のアーキテクチャが採用されている．一方，情報交換や知識共有に関する分野では，サーバを持たずクライアント同士がピア・ツー・ピアで通信するマルチエージェントのアーキテクチャに関する研究開発が盛んに進められている．マルチエージェントのアーキテクチャは，クライアントサーバ型のアーキテクチャと比べ，コミュニケーション範囲の分散協調管理機構を実現するのに適している．このことから我々は，Shine をマルチエージェントシステムとして設計するアプローチを取る．

我々のアプローチでは，コミュニケーションの管理機構

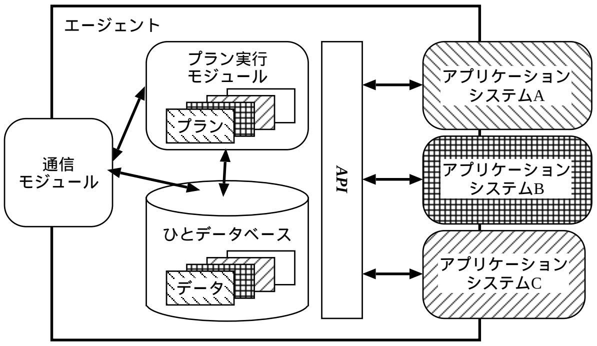

図 1: Shine エージェントの内部構造

やデータは各人のもとに分散配置される．つまり，Shine フレームワークは，各ユーザに対しそのユーザの社会的インタラクションに関わるパーソナルエージェントを提供する．Shine のパーソナルエージェントは他のパーソナルエージェントとピア・ツー・ピアの情報交換を行い，各ユーザに見合ったコミュニティを柔軟に構成していく．

3 Shine エージェントの内部構造

第 2 章で述べたように，Shine ではユーザ毎にパーソナルエージェントが提供される．図 1 は Shine エージェントの内部構造の概念図である．Shine エージェントの内部には，ひとデータベース，プラン実行モジュール，および通信モジュールというシステムの核となるモジュール群が存在する．また，一つのエージェント上に一つまたは複数のアプリケーションシステムが載る．アプリケーションシステムは各モジュールの機能をアプリケーションプログラミングインタフェース（API）を通じて利用し，それぞれのサービスをユーザに提供する．

Shine では，様々なアプリケーションシステムが同一の API を利用してそれぞれのサービスを実現する．その結果，同じ API を使うソフトウェア部品をライブラリとして複数のアプリケーションシステムの間で共有，再利用することができる．

以下，各モジュールについて詳説する．

3.1 ひとデータベース

ひとデータベースは，人およびパーソナルエージェントに関するデータを保持する．保持されるデータには，エージェント自身やそのエージェントが対応しているユーザの情報と，そのエージェントが知っている範囲の他のエージェントおよびそれらが対応しているユーザの情報が含まれる．これらのデータはエージェント間通信により随意流

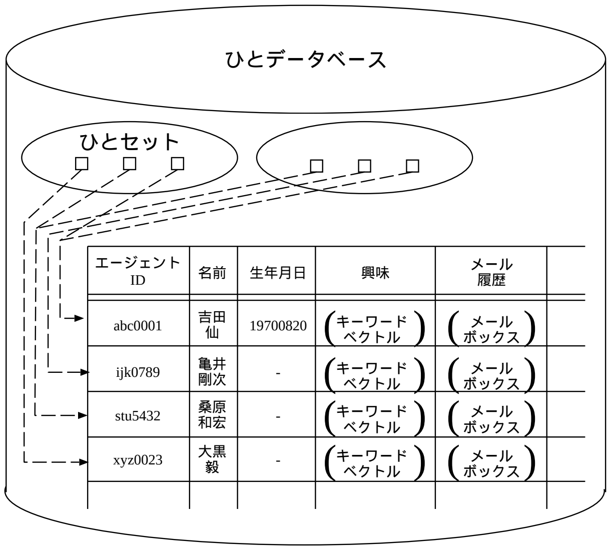

図 2: ひとデータベース

通される．

Shine のアーキテクチャでは，ユーザとそのユーザのパーソナルエージェントは一対一に対応する．このため，ひとデータベースには，人のデータとその人に対応するパーソナルエージェントのデータが区別なく格納される．人のデータには例えば名前や生年月日などがあり，パーソナルエージェントのデータにはエージェント ID やエージェント間通信のためのアドレスなどがある．各アプリケーションシステムはこれらの情報を参照してサービスを行い，また必要に応じて情報を更新する．

ひとデータベースにおいて，「名前」や「生年月日」，「エージェント ID」などはデータの属性と呼ばれる．また，各属性のそれぞれのデータは属性値と呼ばれる．以下では，属性の集合を A，ある属性 $a \in A$ の属性値を $x_a \in X_a$ と記す．

Shine のひとデータベースは，人と属性の組から属性値へのマッピングを行う表を持つ．図 2 にこの表の例を示す．表の各行は，一人の人間，およびその人に対応しているエージェントを表す．また各列は一つの属性を表す．つまり，ひとデータベースに格納されている人の集合を P とすると，この表の一つの列は P から X_a へのマッピング $f_a : P \to X_a$ である．このマッピングを用いて任意の人 $p \in P$ の属性 a の属性値 $f_a(p) \in X_a$ を求める機能が API として提供される．

ひとデータベースに格納されるデータの形態は様々である．エージェント ID やユーザの名前などの単純な数値や文字列のデータの他に，複雑な形態を持つものも含まれる．例えば，情報検索の分野で人の興味を概括的に表すのに用いられるキーワードベクトルや，ある人から受け取っ

たメールを溜めておく「メールボックス」なども，人に関するデータとしてデータベースに格納することが考えられる．Shine のひとデータベースは，こうした様々な形態のデータも，構造化された属性値として表に格納できる．

表に格納される属性のうち，値から人もしくはその人に対応するエージェントを特定することが出来るものはキー属性と呼ばれる．つまり，

$$X_{\mathrm{key}} = \{f_{\mathrm{key}}(p) \mid p \in P\}$$

に対し，$f_{\mathrm{key}}^{-1} : X_{\mathrm{key}} \to P$ という逆のマッピングが存在し，

$$x_{\mathrm{key}} = f_{\mathrm{key}}(p) \Leftrightarrow p = f_{\mathrm{key}}^{-1}(x_{\mathrm{key}})$$

というように属性値と人が一対一に対応するような属性 key がキー属性である．例えば「エージェント ID」はキー属性であり，その値は，エージェント間通信における送信者，受信者の特定や，ひとデータベースの内容のファイルへの書出し及びファイルからの読込みにおける識別子などに用いられる．API としては，属性値 $x_{\mathrm{key}} \in X_{\mathrm{key}}$ を持つ人 $f_{\mathrm{key}}^{-1}(x_{\mathrm{key}}) \in P$ を検索する機能が提供される．

Shine では，人から属性値へのマッピング $f_a : P \to X_a$ も，キー属性値から人への逆マッピング $f_{\mathrm{key}}^{-1} : X_{\mathrm{key}} \to P$ も，ハッシュを用いて高速に検索できるようにする．これにより，例えば「名前」というキー属性の値が 吉田 である人の「興味」を引く，つまり $f_{\text{興味}}(f_{\text{名前}}^{-1}(\text{吉田}))$ を求めるといったことが簡単かつ高速にできる API が提供される．

ひとデータベースはイベント発生機能を持つ．ある人の属性値が追加，変更，または削除されたときには，イベントが発生する．プラン実行モジュールなどの他のモジュールがこれらイベントの発生を監視し，それらに呼応して動作する．このイベント通知機構は，複数のモジュール間での協調や，さらには複数のアプリケーション間の連携を可能にする．例えば，あるエージェントからそのユーザの興味の変化の通知を受け，それに相当するキーワードベクトルが変更されると，その変更を知らせるイベントが発生し，それを検出したアプリケーションシステムが表示画面を更新するといった具合である．

ソーシャルウェアはコミュニティを対象とするので，Shine フレームワークは，個々のエージェントだけでなく複数のエージェントが形成するコミュニティも明示的に扱えるような仕組みを必要とする．コミュニティを扱うために，Shine のひとデータベースは各コミュニティに対し図 2 に示すような「ひとセット」を定義し，個人情報の集合を対象にした操作を容易にする．つまり，あるコミュニティのメンバの集合 $S \subseteq P$ を，そのコミュニティのメンバであるかどうかを判定する関数 $m : X_a \times X_b \times \cdots \to \{\mathrm{true}, \mathrm{false}\}$ を用いて，

$$S = \{p \mid m(f_a(p), f_b(p), \ldots) = \mathrm{true}\}$$

のように取り出せる仕組みを用意する．ひとセットは，メンバの判定に影響を与える属性 $a, b, \cdots$ の属性値が更新されたことを知らせるイベントを受けてメンバを更新する．この機構はコミュニティの構成の動的な変化への追随を可能にする．

ソーシャルウェアのアプリケーションシステムの多くは，メッセージをコミュニティのメンバーに同報する機能を必要とする．ユーザ集合を興味で区切るクライアントサーバ型のアーキテクチャと異なり，コミュニケーション範囲を分散協調管理するアプローチを取るアーキテクチャでは，同報の範囲の決定が自明ではない．Shine エージェントは，同報における宛先のリストにひとデータベースのひとセットを用いることにより，同報の範囲の決定を柔軟に行える．

3.2 通信モジュール

通信モジュールは他のエージェントの通信モジュールとピア・ツー・ピアで接続され，互いにメッセージを交換する．

ソーシャルウェアの様々なアプリケーションシステムでは，それぞれが前提としている通信環境も多種多様である．Shine の通信モジュールは，このような通信環境の違いを吸収し通信モジュール以外のモジュールやアプリケーションシステムに影響を与えないようにするために，図 3 に示すように下位層の通信プロトコルに依存する部分としない部分に分かれている．下位層通信プロトコルに依存する部分をチャネルと呼ぶ．チャネルは，送受するメッセージの，Shine 内での形式と下位層通信プロトコルに合った形式との相互変換を行い，また接続の確立や切断を管理する．

チャネル間の通信は，下位層のネットワークのトポロジによっては，必ずしもピア・ツー・ピアではなくルータ等を介した集中型となる場合もありうる．しかしこれはチャネル層での物理的なネットワークトポロジが集中型であるという事であり，Shine エージェント間通信のレベルの論理的なトポロジはあくまでピア・ツー・ピアの構成をしている．したがって，他エージェントのアドレスの管理等は各エージェントに任されている．

他のエージェントのアドレスは，ひとデータベースに保持されている．プラン実行モジュール等の他モジュールは，送信したいメッセージを作成し，ひとデータベー

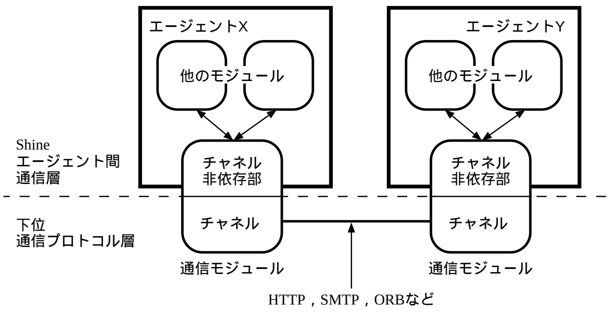

図 3: チャネル

スに格納されている人 $p \in P$ をその宛先としてメッセージに付加する．通信モジュールのチャネルは p からその下位層通信プロトコルにおけるアドレスをひとデータベースを使って $f_{\text{アドレス}}(p)$ のようにして調べ，得られたアドレスに送信する．そのとき，自身のアドレス $f_{\text{アドレス}}(\text{自分})$ を送信者アドレスとしてメッセージに付加する．メッセージを受信したエージェントのチャネルは，送信者アドレスからそのアドレスに対応するエージェントのひとデータベースに於けるエントリを $f^{-1}_{\text{アドレス}}(\text{送信者アドレス})$ のようにして引き，それを付加したメッセージをプラン実行モジュールに渡す．

アドレスの形式は下位層通信プロトコルによって異なるので，チャネルがアドレスを格納するのに使う属性もそれに伴って変わる．例えば下位層通信プロトコルとして HTTP を用いるチャネルは URL アドレスという属性を使い，SMTP を用いるチャネルはメールアドレスという属性を使う．

3.3　プラン実行モジュール

エージェントの行動の規則を記述したものをプランと呼ぶ．Shine では，複数のプランが，プラン実行モジュール内において並行に動作する．これらのプランの中には，各アプリケーションがそれぞれのサービスを実行するために独自に用意するプランと，Shine がフレームワークとして提供する，基盤的なタスクを行うプランがある．

各プランは，色々な事象をきっかけとして，それに呼応するように行動する．動作のきっかけとなる事象には，他のエージェントからのメッセージの受信，ひとデータベースに保持されている値の更新，ユーザからの入力などがある．

プラン実行モジュールは，通信モジュールから渡される受信メッセージを，適切なプランに分配する機能を持つ．各メッセージが，その内容の記述の他にメッセージタイプを指定するフィールドを持っており，メッセージの分配はそのメッセージタイプに従って行われる．このメッセージ分配機構は，各プランが特定のメッセージタイプのメッセージが受信されたときに起動されるメッセージハンドラをプラン実行モジュールに登録することによって成される．

ひとデータベースに格納されている値の更新は，その更新を伝えるイベントを各プランが受け取ることによって，エージェントの動作に反映される．この仕組みを用いて複数のプランが連携することも可能である．すなわち，あるプランがある人の属性値を変えると，別のプランは，それが変わったことによって発生するイベントを受け取ることによって，属性値の変化に対応する．

ユーザからの入力の受付に関しては，各アプリケーションシステムがそれぞれのユーザインタフェースを用意し，それをアプリケーション独自のプランと結び付けることで成される．

4　アプリケーションシステムへの Shine の適用例

この章では，ソーシャルウェアのアプリケーションシステムへの Shine フレームワークの適用について，具体例を用いて説明する．題材として著者らが開発している二つのシステム Community Organizer と「ひとのあかり」を取上げる．我々はこれら二つのシステムおよび Shine をプログラミング言語 Java[1] を用いて実装している．これら二つのシステムの実装において Shine の枠組みが上手く当てはまりフレームワークとして有効に機能することを示す．

4.1　Community Organizer

Community Organizer は，新たなネットワークコミュニティの形成を支援するためのコミュニケーションシステムである [4, 3]．このシステムは，興味を共有する人々をコミュニティの潜在的なメンバと見做し，それらの人々が出会い会話することができる場を提供する．

図 4 は Community Organizer が各ユーザに提供する画面の例である．図中の表示パネルは，コミュニティの潜在的なメンバが出会い会話するための場として提供されるユーザインタフェースである．このパネルには，ベクトル空間モデルにおける各キーワードを軸とする多次元空間を，ある手法により 2 次元平面に射影したものが表示される．ユーザはこの表示パネル上にアイコンを置くことができ，それは他のユーザの表示パネル上にも現れる．各アイコンにはキーワードベクトルが付与され，その値に従って

[1] Java は Sun Microsystems 社の登録商標である．

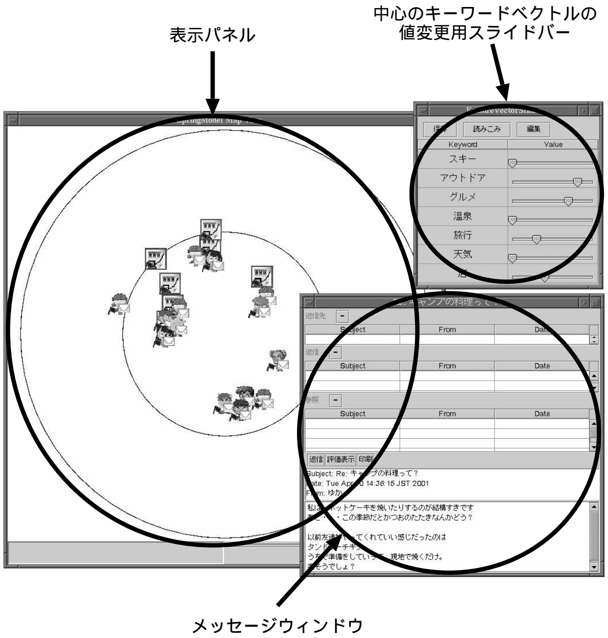

図 4: Community Organizer の画面表示例

計算されるパネル上の点にアイコンが表示される．このとき，パネル上でのアイコン間の相対的な距離は，それぞれのキーワードベクトル間の類似度をもとに，似たキーワードベクトルを持つアイコン同士は近くなるように決定される．

表示パネルにおける中心点は，それに対応するキーワードベクトルを持っており，それは表示パネルの横に設置された一連のスライドバーの位置で示される．ユーザはこのスライドバーを操作することによって中心のキーワードベクトルを変えることができる．中心のキーワードベクトルが変わるとそれにつれて表示パネル全体のアイコンの配置も移動する．また，ユーザが新たに置くアイコンに付与されるキーワードベクトルには，中心のキーワードベクトルの値が設定される．

ユーザは，潜在的なコミュニティを見つけるために，中心のキーワードベクトルをスライドバーで変化させてベクトル空間内を「見てまわる」．そして，例えば “旅行” と “食べ物” の値が大きくなるようスライドバーを動かし，そのようなキーワードベクトルを持つアイコンを新たに置くと，「私は旅行と食べ物に興味がある」ということを他人に表明することになる．つまり，Community Organizer の表示パネルは，興味の表明を表す「興味アイコン」が詰まったベクトル空間を可視化する．

さらに，ユーザがメッセージを書き，それを興味アイコンに添付できるようなインターフェースが提供されている．他のユーザはアイコンをクリックすることでそのアイコンに添付されたメッセージを読むことができ，それに対する返信を添付した新しいアイコンをそのアイコンの近くに置くことによって返事をすることができる．

我々は当初，Community Organizer を Shine フレームワークには依存しない独立したシステムとして開発した．しかし，以下に述べるような方法で，Community Organizer を Shine フレームワークに合わせて再構築することができる．

Community Organizer における興味アイコンは，それを置いたユーザ，それが持つキーワードベクトル，および添付されたメッセージという情報を持っている．これを Shine のひとデータベースに格納するときには，各属性値がリスト構造になっている「興味アイコンリスト」という属性を用意する．そしてキーワードベクトルと添付されたメッセージをひとまとめにした形態のデータを，「興味アイコンリスト」属性の列の，そのアイコンを置いたユーザに相当する行にあるリストの中に格納する．

興味アイコンを画面に表示するときには，各ユーザの興味アイコンのリストの要素を「興味アイコンリスト」属性の列から取りだし，「アイコン画像」属性の列にあるユーザを表すアイコンの画像ファイルを用いて表示する．

Community Organizer の表示パネルには，データベースに格納されている全ての興味アイコンのうち，中心のキーワードベクトルに近いキーワードベクトルを持つものだけが表示される．この選択に，ひとデータベースのひとセットを用いることができる．すなわち，メンバを判定する関数が，中心のキーワードベクトルと各興味アイコンのキーワードベクトルの間の余弦尺度の値のうち上位に来るものだけを選ぶものであるようなひとセットを用意し，そのメンバをパネルに表示する．

元々の Community Organizer は，他の多くのソーシャルウェアアプリケーションシステムと同様，クライアントサーバ型のアーキテクチャで設計されていた．しかし，第 2 章で述べたように，Shine は柔軟なエージェント指向アーキテクチャを採用している．Shine フレームワーク適用後の Community Organizer においては，サーバは存在せず，各クライアントプログラムはサーバが持っていた機能の一部が組み込まれエージェント化される．それらエージェントは，Community Organizer の動作に必要な個人情報を他のエージェントとピア・ツー・ピアで交換する．

プラン実行モジュール内の Community Organizer プランは，ユーザが新たに作成した興味アイコンを他のエージェントに送信し，また他のエージェントから送られてきた興味アイコンをひとデータベースに格納する等の動作をする．

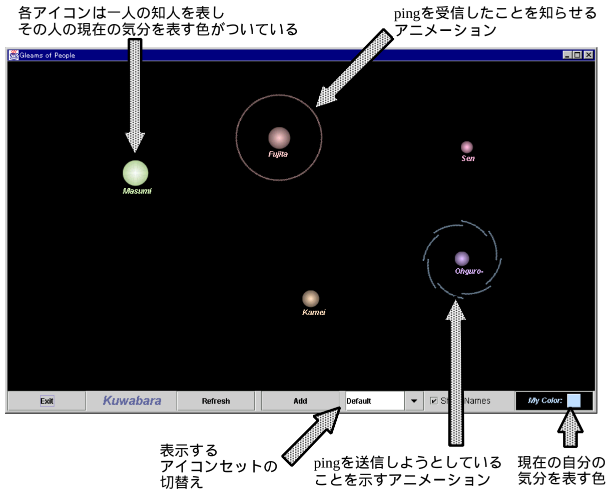

図 5: 「ひとのあかり」の画面例

4.2　ひとのあかり

「ひとのあかり」は，コミュニティのメンバが互いに存在や気分を伝えあうことにより，コミュニティ感覚を維持し発達させるための，従来になくシンプルなメディアによるコミュニケーションシステムである [6]．

「ひとのあかり」の機能を直感的に説明すれば，人間を対象とする ping コマンドであるといえる．つまり，人同士で ping パケットを交換することで互いの存在を伝えあう．また，単なる存在伝達に加えて，「ひとのあかり」で交わされるパケットは送信者の現在の気分を色で表現した情報も伝える．図 5 は「ひとのあかり」の画面表示例である．

このシステムの動作は以下のようになる．まずユーザが自身の現在の気分を表す色を選択する．また，対象とする人を数名指定する．指定された人は画面内に球状のアイコンとして表示される．あるアイコンをダブルクリックすると，アイコンに向かって同心円が縮んでいくアニメーションが行われ，次いでそのアイコンに対応する人のパーソナルエージェントに ping が送られる．パケットには，送信者の現在の気分を表す色の情報が付加される．逆に，ある人から ping が送られてくると，今度はその人を表すアイコンから同心円が拡がっていくようなアニメーションが行われ，またそのアイコンの色が ping が伝えてきたものに変わる．同時に，送られてきた ping への返信の ping が，今の自分の気分として設定されている色と共に自動的に送られる．

「ひとのあかり」も Community Organizer と同様に Shine とは独立に作られた．元のシステムは，個々のユーザに対応するパーソナルエージェント群と，リピータと呼ばれるメッセージ交換装置から成っており，メッセージは全てリピータを介して送られる集中型のネットワークトポロジとなっている．リピータは，あるエージェントがメッセージを送ろうとした際，宛先のエージェントがメッセージを受け取れる状況でなかった場合に，送られるべきメッセージを一時的に溜めておき，後にメッセージを受け取れるようになった時点で配送する機能を持つ．

このような「ひとのあかり」のメッセージ交換機構を Shine フレームワークに当てはめた場合，リピータはチャネルに隠蔽される．エージェント間のネットワークの論理的なトポロジは，下位通信プロトコル層ではリピータを中心とする集中型，エージェント間通信層ではピア・ツー・ピア型となる．また，チャネルを，リピータを介さないピア・ツー・ピア型のものに置き換えることも容易に出来る．

ひとデータベースには，名前，色，メールボックスといった属性を持たせる．あるユーザが別のユーザへ ping を送信したときの，それぞれのひとデータベースにおける値の更新は次のように行われる．送信者エージェントのひとデータベースにおける送信者のエントリを p，宛先のエントリを q，宛先エージェントのひとデータベースにおける送信者のエントリを $p\prime$，宛先のエントリを $q\prime$ とする．送信者が，画面上の宛先のアイコンをクリックすると，ユーザインタフェースはそのイベントを受けて送信者の気分を表す色 $c = f_{色}(p)$ を宛先 q に送るよう通信モジュールに依頼する．通信モジュールは，それを宛先エージェントの通信モジュールに届ける．これを受け取った宛先エージェントの通信モジュールは，送信者のアドレスから，宛先エージェント内のひとデータベースにおける送信者のエントリ $p\prime$ を割出し，c と共にプラン実行モジュール内の「ひとのあかり」プランに伝える．プランは，$p\prime$ の色 $f\prime_{色}(p\prime)$ を c に更新する．この更新はユーザインタフェースに伝わりアイコンの色が変えられる．同時に，送られてきた ping メッセージに対する返事が，$q\prime$ の気分を表す色 $f_{色}(q\prime)$ と共に $p\prime$ に送られる．

「ひとのあかり」では，表示するアイコンを何個かずつセットにして切替えることができる．アイコンのセットは，ひとデータベースのひとセットをそのまま対応させることができる．

4.3　二つのアプリケーションシステムの Shine 上での実現に関する考察

我々の実装において，Shine フレームワークの Java クラスファイルの合計，Shine 部分とその他に既存のクラスライブラリを利用している部分を除いた Community Organizer 自体の Java クラスファイルの合計，および Shine 部分を除いた「ひとのあかり」自体の Java クラスファイルの合計を比較した結果，Community Organizer では全体の約 19% が，「ひとのあかり」では約 53% が Shine フレームワークによって提供された．

ところで，ソーシャルウェアのためのフレームワークを提供することの目的の一つに，アプリケーションシステム間で連携を可能にすることがある．Community Organizer と「ひとのあかり」でも，一つの Shine エージェント上で両者を並行に動作させることができる．このとき，ひとデータベース，通信モジュール，プラン実行モジュールは一つのものが共有される．ひとデータベース内のデータに関しては，各アプリケーションシステムが独自に属性を持つ他，共通の属性を持つことによって両システムを連携させせることができる．例えば「ひとのあかり」の ping の頻度を「親しさ」のパラメータとして Community Organizer と共有し，Community Organizer はそれを表示パネルにおけるアイコンの配置に利用するといったことが考えられる．通信モジュールについてはチャネルも含めて共有される．交換されるメッセージの各アプリケーションシステムへの振り分けは，両者が別々のメッセージタイプをメッセージに付加することにより成される．

5　おわりに

本報告では，ソーシャルウェアのエージェント指向フレームワークについて論じ，著者らが開発しているフレームワーク Shine の具体的な構成について述べた．また，応用例として，Community Organizer と「ひとのあかり」への Shine の適用について述べた．これら二つのシステムの実装において Shine の枠組みが上手く当てはまりフレームワークとして有効に機能することを示した．

Shine には多くの課題が残されている．プラン実行モジュールにおけるプランの動作については，幾つかのプランを実際に作成することを通じ，より詳細な機能要件を明らかにしていく必要がある．また，プライバシ管理機構についても取り組まなければならない．さらに，Community Organizer と「ひとのあかり」の間のデータ共有による連携を実装し，アプリケーションシステム間連携を実証したい．

参考文献

[1] F. Hattori, T. Ohguro, M. Yokoo, S. Matsubara, and S. Yoshida. Socialware: Multiagent systems for supporting network communities. *Comm. ACM*, 42(3):55–61, 1999.

[2] T. Ishida ed. *Community Computing — Collaboration over Global Information Networks.* John Wiley & Sons, 1998.

[3] K. Kamei, E. Jettmar, K. Fujita, S. Yoshida, and K. Kuwabara. Community organizer: Supporting the formation of network communities through spatial representation. In *Proc. 1st SAINT 2001*, pp. 207–214. IEEE Computer Society, 2001.

[4] 亀井, 吉田, 大黒, 服部. Community Organizer: ネットワークコミュニティの形成支援. ヒューマンインタフェースシンポジウム'99 論文集, pp. 333–336, 1999.

[5] 岸, 野田. ソフトウェアアーキテクチャ. コンピュータソフトウェア, 18(2):68–77, 2001.

[6] T. Ohguro, S. Yoshida, and K. Kuwabara. Gleams of People: Monitoring the presence of people with multi-agent architecture. In *Approaches to Intelligent Agents — Proc. 2nd PRIMA 1999*, Vol. 1733 of *LNAI*, pp. 170–182. Springer-Verlag, 1999.

[7] 梅木. ネットワークコミュニティ形成支援技術. 人工知能学会誌, 14(6):943–950, 1999.

[8] S. Yoshida, K. Kamei, T. Ohguro, K. Kuwabara, and K. Funakoshi. Building a network community support system on the multi-agent platform Shine. In *Design and Applications of Intelligent Agents — Proc. 3rd PRIMA 2000*, Vol. 1881 of *LNAI*, pp. 88–100. Springer-Verlag, 2000.

[9] S. Yoshida, T. Ohguro, K. Kamei, K. Funakoshi, and K. Kuwabara. A platform for making network community support systems in a cooperative distributed architecture. In *Proc. 7th ICPADS 2000: Workshops*, pp. 441–446. IEEE Computer Society, 2000.