SAGE (Smart AGent Environment)

— メッセージ指向のエージェント環境へ —

佐藤　陽† 益岡　竜介† 菅坂　玉美† 北島　弘伸† 丸山　文宏†

Akira SATO†, Ryusuke MASUOKA†, Tamami SUGASAKA†, Hironobu KITAJIMA†, and Fumihiro MARUYAMA†

あらまし
我々は, 情報の公開や公開された情報の積極的な利用, 情報の共有などの問題の解決に知的エージェント環境 SAGE (Smart AGent Environment) [1] の適用を試みており, SAGE: 仮想カタログ [2] [3] では, 項目や値の種類の異なる異種のデータベースを統合し, 一つの仮想的な知識ベースとして見せることに成功している.
この SAGE のアーキテクチャをより広範囲に適用していくためには, 幅広い要求に対応できる, より汎用で応用範囲の広い枠組みが望まれる. 具体的には高速な応答性, 負荷の分散や知識の分散のしやすい構成, 全体系の設計の変更や拡張が容易に行なえ異なる分野にも適応しやすい分かりやすいアーキテクチャ, 管理や保守の簡易さなどが求められると考える.
本稿では, こうした効率, 設計, 管理などの問題により適した枠組みとして, メッセージの状態遷移を中心に考えたアプローチ (メッセージ指向と呼ぶ) を検討する.

Abstract
We are currently working on applying the SAGE (Smart AGent Environment) [1] to publishing, utilizing and sharing information distributed over networks. We have already succeeded in seamless integration of heterogeneous databases and turning them into a single virtual knowledge-base system. [2][3]
In order to extend the scope of its application, we need a more generic framework to meet various kinds of requirements from miscellaneous domains. Such flexible system must possess responsiveness as a total system, adaptive load-balancing, highly distributed architecture for processes and knowledge, and ease of administration whenever needs arise.
In this paper, We introduce and describe a new message-central approach which can address above issues by means of state-transition of messages.

† (株) 富士通研究所, 福岡市
Fujitsu Laboratories LTD., 2-2-1 Momochihama Sawara-ku, Fukuoka-shi, Fukuoka, 814, Japan

1. はじめに

WWW の成功などにより, 情報の公開や公開された情報の積極的な利用が最近急速にすすんできている. それに伴い, 公開された情報を共有したり, 統一した形式で扱うような動きも積極的である. 特にビジネスの分野では, 独自にデータベース (以下 DB) を持つ企業も増え, DB の規模も相当の大きさになってきており, DB の情報公開や, 近い業種間での情報共有などを実現しようとする意識が高まっている. しかし, 企業などの各組織ごとに独自に管理運営している DB では, その提供者ごとに構造や扱っている用語が異なるために, 形式を統一したり管理したりすることには手間や困難が伴う. 我々は, こうした問題の解決に知的エージェント環境 SAGE (Smart AGent Environment) [1] の適用を試みており, SAGE: 仮想カタログ [2] [3] では, 項目や値の種類の異なる異種のデータベースを統合し, 一つの仮想的な知識ベースとして見せることに成功している.

SAGE では, 情報の要求者や提供者など (ユーザやデータベースなど) を知識をやり取りするエージェントと呼ばれるソフトウェアにしている. それらの間の情報交換をうまく設計し, 仲介エージェント (ファシリテータ) と呼ばれるエージェントを介して各エージェントがやりとりすることで異種の情報の統合や情報交換を円滑に行なうことができる.

ファシリテータにはその機能として, メッセージに合った適切なエージェントを選びだしてその選択したエージェントのみにメッセージを転送する機能や, 受け取りのエージェントに合わせてメッセージの形式やその中で使われている言葉などの中身を変換する機能がある. ネットワーク上に分散し提供者ごとに形態の異なるもの同士をつなぐためには, これらの機能は本質的に必要な技術である.

我々は, このファシリテータを中心とした SAGE のアーキテクチャを, より広範囲に適用し, 情報の共有や統合をより広い範囲で行なったり, 情報の統合検索以外の分野に適用していくことのできる枠組みとしてとらえている. そうした応用の中には, 異業種を結び付けるために複数のファシリテータが結び付き, ファシリテータ同士でもさらに互いの情報を仲介する仕組みが必要となるような状況や, 逆にファシリテータの一部機能だけを利用し, そのかわり高速なレスポンスを要求するような状況も想定される. それら幅広い要求に対応できるより汎用で, 応用範囲の広い枠組みを目指すためには, (1) 高速な応答性, (2) 負荷を分散したり, 異なる組織ごとに別のシステムを用意する必要がある場合などに適応できる分散に強い構成 (知識を分散的に管理することと処理を分散することを共に含む) (3) 異分野への適用や, 全体系の細かなバージョンアップに適応できる変更や拡張に強く, 内部構造の分かりやすいアーキテクチャ (4) 管理や保守の簡易さなどが求められると考える.

本稿では, こうした効率, 設計, 管理などの問題をメッセージのやりとりをどのように定義し, 設計し, 管理するかという問題ととらえ, その問題により適した枠組みとして, メッセージの状態遷移を中心に考えたアプローチ (メッセージ指向と呼ぶ) を検討する. 特にメッセージの効率の点からは, メッセージ指向に基づいたメッセージの配送例を示し, また設計や管理の面からのメッセージ指向のアプローチの意義についても述べる.

2. 従来のメッセージ配送

SAGE: 仮想カタログは, ユーザインタフェースをエージェント化したユーザエージェント, 各データベースをそれぞれエージェント化したデータベースエージェント,2 者を仲介するファシリテータの 3 種類のエージェントから成るシステムである. エージェント間のメッセージのやりとりには ACL (Agent Communication Language) [4], [5], [6] を用い, 以下のような流れで検索を行なう (図 1).

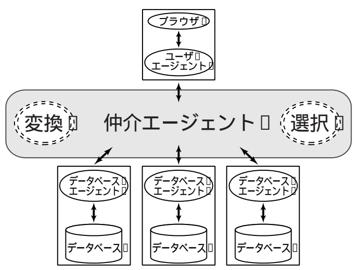

図 1　従来のメッセージの配送

(1)　まずユーザエージェントが, ユーザの入力を対応する ACL に変換してファシリテータに問い合わせる.

(2)　ファシリテータで ACL メッセージを受けると, 入力された問い合わせに関する情報を持っているエー

ジェントを選びだし, 各データベースエージェントが扱えるように項目名や単位, 問い合わせのしかたなどをそれぞれのエージェントに合わせて変換し, それを各データベースエージェントに仲介する.

(3) 各データベースエージェントでは, ACL をデータベース用の問い合わせ (SQL など) に変換して実際の検索を行ない, その結果をまた ACL に変換してファシリテータに返す.

(4) ファシリテータは, 返ってきた回答に関しても (データベースごとに言葉の体系が異なるので) ユーザエージェントの求める言葉の体系に変換する. また, 各データベースエージェントに送った問い合わせの回答結果を全てまとめる. そして一つのメッセージにしてユーザエージェントに返す.

(5) ユーザエージェントは返ってきた回答をユーザに分かりやすい形式で提示する

このようにして, SAGE: 仮想カタログのシステムでは, ネットワーク上に分散的に配置される複数の商品情報などのサービスを一つの仮想的なカタログとして見せることを実現している. そして, 実現にあたって, ユーザエージェントとデータベースエージェントとの間にファシリテータを用意する構成をとり, このファシリテータにユーザエージェントの必要とする内容に基づいてデータベースエージェントを選択する機能や, エージェント間の用語体系の違いを吸収する変換機能を持たせ, さらに複数のデータベースエージェントから返ってくる回答を統合する機能も持たせるという構成をとっている.

3. メッセージに着目した課題

前節で説明したファシリテータを介してメッセージをやりとりする方法では, ファシリテータにサービスを提供するエージェントを選択する機能, エージェント間の用語の違いを吸収する機能, データベースエージェントの返す各回答に関してユーザエージェントに合わせた変換を施す機能, 返ってきた複数のエージェントからの回答を統合する機能などを持たせている. これらの機能は異種の情報を扱い, 統合するシステムの性格上必須のものである.

しかしながら, これらの機能をすべて一つのファシリテータに持たせていることにより, 以下のような問題点もある.

(1) 対応するエージェントを選びだす機能や選びだされた各エージェントとの間のやり取りを行なうためのメッセージの変換, 複数のメッセージの統合機能などがすべてファシリテータに集中することにより, ファシリテータの負荷が増大し, エージェント間で大量のやりとりが起こる場合に応答性が悪くなるなどの可能性がある.

(2) 特に検索結果を返す場合に, 回答メッセージが一つで統合する必要がない場合や, 回答を別の形式に変換する必要がない場合にも必ずファシリテータを経由するようになっているため, ファシリテータを必要としない場合にも経由しなければならず, 経路的に無駄がでる.

このうち (1) は, 主にエージェントの負荷に関する問題であり, (2) はメッセージ配送の効率に着目したシステム全体の応答速度につながる問題である. しかし, 例えば (1) においてやりとりの数が増えると, 一つ一つのメッセージの応答性が悪くなることも考えられるため, (1) もシステム全体の応答性に影響を与える可能性がある. SAGE: 仮想カタログのシステムにおいては, ネットワークの混み具合いなどによっても多少異なるが, 現在のエージェント間のメッセージの送受信には 1 回につき約 0.2 秒かかっている(注1). 今後の応用によっては, ファシリテータを複数経由するような構成も考えられ, そのような場合にはメッセージの送りかたを考えないと送受の回数が増えてシステム全体としての応答時間も単調に増えることになる. そこで, 一つ一つのメッセージのやりとりの効率を改善することはもとより, メッセージの効率の良いやりとりを考えることが, システム全体の効率を考える上で重要なポイントとなってくる.

このうち特に (1) の問題点を解決するために, ファシリテータの中の複数の機能を分割して別のエージェントにすることも考えられる (図 2). 仮想カタログにおいては, 目的にあったエージェントを選びだす機能を別エージェントにしたり, メッセージの翻訳を行なう機能を別のエージェントにするなどのことを考えることができる. 特に大きなデータベースを必要とするような翻訳などの機能については外部のエージェントに分けるのは有効であるように思われる.

このような分割を行なった場合には, 確かに機能的には分散されるため, 各エージェントの抱える負荷の問題は軽減される. しかし, 同時に以下のような問題

(注1): 送り側のメッセージの生成や, 受け側の解析に伴う時間も含めるとより多くの時間がかかっている

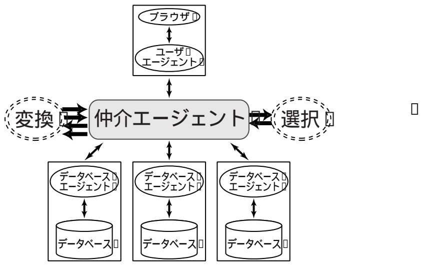

図 2　ファシリテータの単純な機能分割

を生じる.

・　例えば,「目的のエージェントを選びだしてそのエージェントにメッセージを送る」という行動を行なうためには, エージェントの選択機能を持つエージェントに対してアクセス先のエージェントの選択を依頼し, 続いて, メッセージの翻訳機能を持つエージェントに対してメッセージの翻訳を依頼してから, その翻訳メッセージをその選択エージェントに渡すという処理を行なわなくてはならない. つまり, ファシリテータ単独で行なっていた処理が, 複数のファシリテータにまたがる処理になってしまっていてまとめて一回で行なうことができなくなってしまう.

・　その結果, まとめ役のファシリテータと各機能を担うエージェントとの間のやりとりが重なることにより, やりとりの数, つまりメッセージの数はかえって増加してしまう.

4.　メッセージ指向アプローチの導入

上記の課題を解決し, 同時に複数のエージェントのやりとりを設計したり管理したりすることを効率よく行なうための枠組みとして, 以下では新しいメッセージ形式とそれに伴うサービスや配送の仕組み (メッセージ指向アプローチ) について説明する.

4.1　メッセージ

メッセージ指向アプローチでは, メッセージを単一のやりとりではなくさまざまなエージェントをたどる間に状態遷移していく一連のやりとりの流れを表すものと考え, かつ, そのやりとりを通じて受ける処理内容をメッセージ自体に記述するものとする. このように定義し, メッセージを中心とした視点でとらえると, メッセージとは, 依頼元から発信された後, メッセージ自体に記述された様々な処理を受けて状態を遷移しつつ最終的な到達先 (依頼元と同一であることも多い) に届くまでの一連の処理の流れとしてとらえることができる.

4.2　メッセージ形式

上記のような性質を持つメッセージを実現するため, メッセージに, どう処理されるかという内容 (サービス) を記述したサービスリストと呼ぶリストを (エージェント間でやりとりするメッセージに通常伝えている内容とは別に) 付加することにする (図 3).

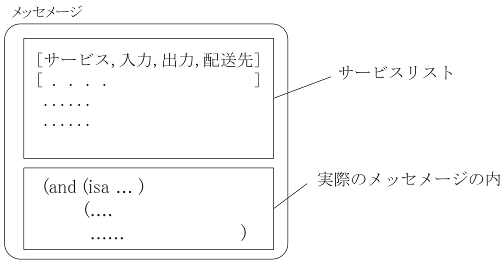

図 3　メッセージの形式

サービスリストは, 配送先のエージェントで実行すべきサービス名, サービス名の指す処理が実行される前のメッセージの状態 (入力のオントロジーなど), サービス名の指す処理が実行された後のメッセージの状態 (出力のオントロジーなど), 配送先のエージェント名を対にしたものを 1 サービスリスト単位として, そのサービスリスト単位がリスト状につながることで構成される. そして, これがメッセージの受ける処理の内容および手順を指示する役割を果たす.

・　サービス名は, 入力状態と出力状態の間の関係を宣言的に記述したものである. この関係の示す内容は, エージェント内の処理, メッセージに施す処理, サービスリスト中のサービスやメッセージの配送経路の再設定や具現化, メッセージの次のエージェントへの送信などに関するどれかもしくは複数の制約が含まれたものと考えることができ, 各エージェントはその制約に従いながら個々のエージェントの仕様や実装に依存した処理を行なうことになる.

どのような物事も適切に準備することによってサービスとなり得るが, 便宜的に例をあげると, データベースの検索, 問い合わせメッセージを対象データベースへ適合するための変換, 適切なエージェントへの配送経路の設定, 複数のメッセージのマージなどをサービスに位置づけることができる.

・　入力状態と出力状態には, サービスが適用される前と後とでメッセージが遷移する状態を記述する.

・　配送先エージェント名には, メッセージを配送し処理する主体となるエージェント名を指定する.

・　サービスリストのこれら各要素の一部を配送過程で未定義 (『*』 で表す) にしておくこともできる. その場合, 途中のエージェントが他の要素に基づいてその部分を詳細化する. また, 特にサービス名については詳細部分まで決定されていない (抽象レベルにある) ときには, その旨を示す記号『*』をその頭につけて表すこととする.

このメッセージ形式では, エージェントの内部に暗黙的に保持されている様々な処理の手順をエージェントの外部にくくり出しているため, 手順どおり進めることを他のエージェントにまかせたり, 別のエージェントが処理の手順を操作したりすることが可能になる. 例えば, ファシリテータがまとめてやっていた処理を他のエージェントで処理したり, 単一のファシリテータが決めていた処理手順を他のファシリテータなどのエージェントで決めることが可能になる.

4.3　メッセージの処理

上記のメッセージを処理するエージェントは, 1 処理 1 エージェントという対応ではなく,1 つのエージェントが複数の処理を同時にこなすこともあることを考慮している.

このメッセージ形式を扱うエージェントの行なう処理 (やエージェントの種類) には, 検索などの従来から行なっている処理の他に, このメッセージ形式を扱ううえで特徴的な処理が考えられる. それらの特徴的な処理のいくつかについて以下で説明する.

4.3.1　サービスの実施

普通のエージェントの行なう処理は, 受け取ったメッセージに自分の持つサービスを適用し, 次へ回すことである. それらのエージェントは, まずサービスリストの一番先頭のサービスを見て, 中に指定されたサービスを実行する. その際, 指定された入力形態と出力形態に合わせてメッセージ中にサービスリストと一緒に記述された (本来の) 中身に必要な変更を施す. そして, 行なったサービスをメッセージ中のサービスリストから削除して (あとのサービスリストだけを残して) 残ったサービスリストの先頭のサービスを実行するエージェント, もしくはそれが記載されていない場合にはファシリテータにメッセージを送信する.

4.3.2　サービスリストの書き換え

メッセージ指向アプローチでは, 単にサービスリストを消費する (サービスを実施する) エージェントだけでなく, サービスリストの書き換えを行なうエージェントを考えている. 書き換えのしかたはいろいろ考えられるが, 主として処理内容をより詳細にしていく (詳細化と呼ぶ) ことを考えている. 例えば, 図 4はサービスリストの書き換え時のサービスリストの特にサービス名の変化を示したものだが, ここでは `*search` という抽象的な『検索』を表すサービスを適切な変換を行なう `translation-1` という具体的なサービス, 抽象的な `*hotel-search` という (ホテルに特化した) 検索サービス, サービスの実施後に適切な変換を行なう `translation-2` という具体的なサービスの 3 つのサービスからなるリストに詳細化している. さらに, このうち `*hotel-search` については, 別のエージェントがより具体的なサービス (リスト) に詳細化するといったことも可能である.

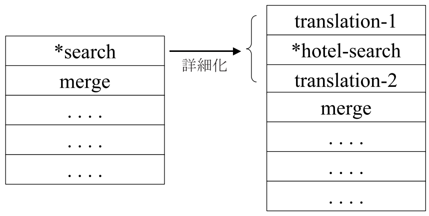

図 4　サービスリストの書き換え

詳細化を行なうエージェントは普通のエージェントとは異なり, サービスリストのトップ以外の部分, 場合によっては全リストに渡って作り直すことのできる知識を持っている. そしてその知識を使って自分以降にメッセージが受けるサービスの内容と順番を, 一部もしくは全てに渡って指定することができる. リストの中身のうち未定義の要素を具体化し, 変換などの処理が間に必要であれば別の内容のリストに置き換えたりリストを増やしてネストしたりしてサービスリストを組み直す. サービスリストの組み直しは, 必ずしも単独の一つのエージェントで完結する必要はなく, 抽象的な状態のままで送り出し, 他のエージェントに詳細化を委ねることもできる. サービスリストの全てが詳細化されていなくとも, リストの詳細化されていない部分に到達するまでは普通のエージェントがそのメッセージを処理することは可能である. これにより, サービスをよりよく知ったエージェントにメッセージを届けて書き換えを任せるようにして, 適切な時点で経路やサービス名を効率のよいものに書き換えたり, メッ

セージの処理がある程度進んだ時点でサービスを具体化したりすることが可能になる.

4.3.3 サービスの横取り

通常のエージェントはメッセージ中のサービスリストの先頭のみを処理して取り除いて次のエージェントに渡すが, ここで述べる『横取り処理』を行なうことにより, エージェントに追加されたり変更された処理を詳細化された全体の流れに変更を与えることなく組み込むことができる.(注2).

横取りを実行するやりかたには 2 通りある.

(1) 自分の本来呼ばれた処理を行なった後, サービスリストをみて次にあるサービスを確認する. もし次のサービスも自分が行なうことができるならば, 他のエージェントに送ることなく自分でそのサービスも処理してしまう. この場合, サービスには違うエージェント名が書かれているのだが, そのエージェントに送らずに勝手に奪ってしまう.

(2) あらかじめサービスリストのはじめの 2 つもしくはそれ以上の要素をまとめてチェックする方法もある.

例えば, データベースを検索するエージェントの行なう処理が, 最初は検索した結果を出力するのみであったのだが, 機能を拡張した結果, 検索の結果を変換する機能も同時に実現するようになった場合を考える. このとき, いったん結果を出してからその結果を変換する処理を行なうのが (1) であるが, 検索と変換を順番に別々に行なうよりは一つの処理にした方が効率がよいなどの理由でまとめて扱うこともできる. その場合には, あらかじめサービスリストとマッチすべきサービスの複数の組み合わせを用意し, それとサービスリストの先頭の複数とをマッチさせる. この際サービスを行なうエージェントの項は (1) と同様無視して扱う. そして, マッチした場合にはまとめた処理を実行する.

マッチしなかった場合にも, 改めてサービスリストの先頭の単一のサービスと本来のサービスとをマッチさせて, 一致した場合には普通の単一の処理を行なう(注3).

この方法によれば, 詳細化された処理の流れとして与えられたサービスリストを変更することなく, 流れに沿った新しい機能を追加することができる. 詳細化が複数の場所で行なわれたり別の組織で行なわれたりして, 必ずしも詳細化のしかたの管理と実際にその詳細レベルに応じた処理を行なうエージェントの管理とが同じ組織で行なわれていない場合, この方法で詳細化に影響を与えずに局所的な最適化などを行なうことができる.

例えばエージェント間のやりとりの統一的な詳細化のしかたが『検索』, 『変換』とを別々にしてあったとしても, 『検索』と『変換』とを同じエージェントで最適化して同時に行なうことができる.

4.4 メッセージ指向導入の利点

上記のようにサービスリストの付加されたメッセージをエージェントからエージェントへの配送していくことで処理を実行していく構成をとることには, 以下のような利点がある.

(1) 1 対 1 ではないメッセージの送信が可能になる. その結果として, サービスを実現するエージェントの選びだしを行なう際には, 単独のエージェントを選びだすだけでなく, 複数のエージェントをまたがって行なう処理全体を選びだすという拡張が可能になる.

(2) エージェントを追加するときには, ファシリテータに登録し, またその追加されたエージェントを使って行なう複合的な処理内容についてもファシリテータの知識に追加するだけでよく, その他のエージェントには変更や追加を行なう必要がない. よってエージェントの追加・拡張や, それに伴なう複合的な処理内容の追加・拡張が容易である.

(3) メッセージの処理状態をメッセージ自体に持たせる構成であることから, 複数のエージェントに渡るような複雑な処理手順を踏む必要がある場合にも, 一つのエージェントがその処理手順のめんどうをみて複数のエージェントにメッセージを順番に配るような処理を行なう必要がない. そのため, エージェントの持つ機能を効率的に分散することができると同時に, 1 つのエージェントに負荷が集中することを回避でき, 負荷の分散を効果的に行なうことが可能になる.

5. メッセージの配送例

メッセージ指向アプローチを導入すると, SAGE: 仮想カタログのメッセージの流れを図 5のようにすることができる.

(注2): 横取りとしてではなく, 最初から複数の処理を同時に行なうエージェントとしておいて, 詳細化の際に複数のサービスをまとめて処理するように詳細化してもらうことも可能である. これは機能分割の粗いエージェントと細かいエージェントとが共存するために必要な要素と考えている

(注3): 本来そのサービスを行なうことになっているので, そのサービス単一でも実行可能でなくてはならない

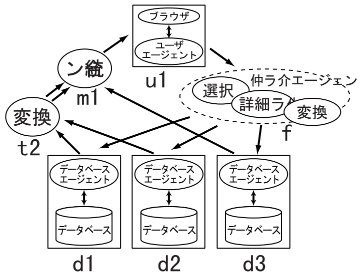

図 5　メッセージ指向の配送例

この例では, ユーザエージェントを u1, データベースエージェントをd1, d2, d3 とする. また, ファシリテータ f はユーザエージェントから受け取ったメッセージを適切なエージェントを選択し, その選択したエージェントに関する全部の経路を設定し, そしてメッセージの中身を変換してそのエージェントへ送り出すところまでを行なうエージェントとする. それとは別に検索結果を変換するエージェント t2 とメッセージの統合を行なうエージェント m1 を用意する.

以下の図 6 ~ 11では, 図の左側と右側にエージェントを通過する前後のメッセージ, 真ん中にエージェントの機能を示すこととする. メッセージのうち上の部分がサービスリスト, 下の部分がメッセージの内容を表す.

(1)　ユーザエージェント u1 がファシリテータ f に問い合わせのメッセージを送る. この時点ではサービスが検索であることのみを指定し, どのエージェントに配送するかなどは未定義である (図 6左のメッセージ). ここでは, *search のように『*』をつけることで詳細なサービスではないことを表している.

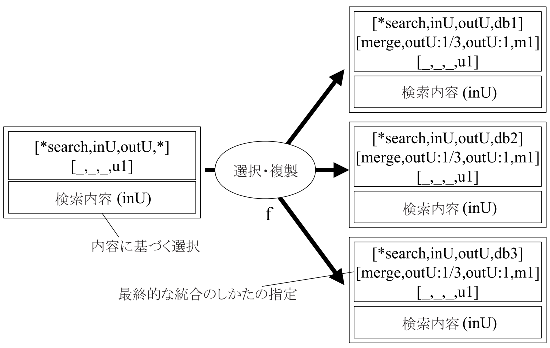

図 6　エージェントの選択

(2)　ファシリテータ f では入力された問い合わせに対応するエージェントをサービスと内容に基づいて判断して選びだす (図 6). そして, 必要なサービスに対応した{ [translate,...], [search,...], [translate,...] } のような詳細なサービスに具体化した内容でサービスリストを書き換え, 検索エージェントを通るサービスリストを持つメッセージを複数 (選ばれた検索エージェント分) 生成する (図 7).

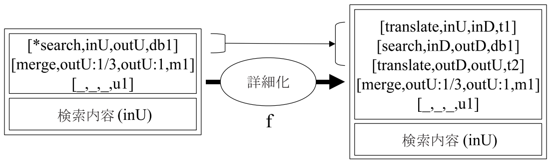

図 7　サービスの詳細化

メッセージが複数の経路に配送される場合には, メッセージ分割値と呼ぶ数字を入力状態と出力状態を記述するところに指定する.　例えば[merge, outU:1/3, outU:1, m1] はエージェント m1 でメッセージのマージ (統合) を行ない, その入力状態ではメッセージ分割値が1/3 であり, 他のメッセージと統合した結果が 1 になることを表している.

(3)　サービスリストの先頭のサービスである変換を行なう (図 8). 一般的には, 変換を行なう別個のエージェントが行なっても良いのだが, この例ではファシリテータ f が変換まで行なっている. 変換を行なった後, 次の経路であるデータベースエージェント (d1,d2,d3) にメッセージを配送する.

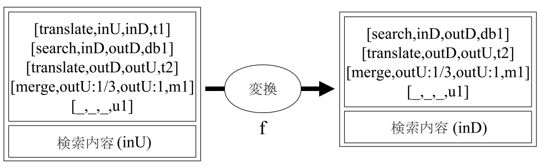

図 8　検索内容の変換

(4)　各データベースエージェント (d1 ~ d3) では, エージェント間メッセージをデータベース用の問い合わせに変換して実際の検索を行なう (図 9). そして, 検索結果を内容としたメッセージを作り, 次の経路である変換エージェント t2 に送る.

(5)　変換エージェントt2 では, データベースエージェントの回答をユーザエージェントの求める言葉の体系に変換して統合を行なうエージェントm1 に渡す (図 10) .

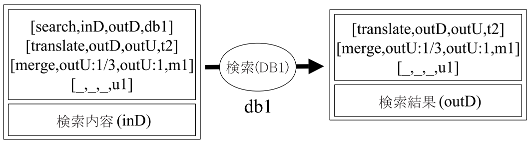

図 9　検索の実行

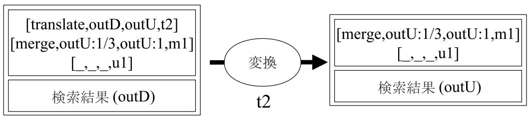

図 10　回答メッセージの変換

(6)　統合を行なうエージェントm1 では, 返ってきた複数の回答をまとめる. 回答が全部まとまったか (到着したか) は, (2) で設定した分割値が出力状態に記述された値に達したかどうかで確認する (図 11) . 回答メッセージの統合後, エージェント m1 はメッセージをユーザエージェントに返す.

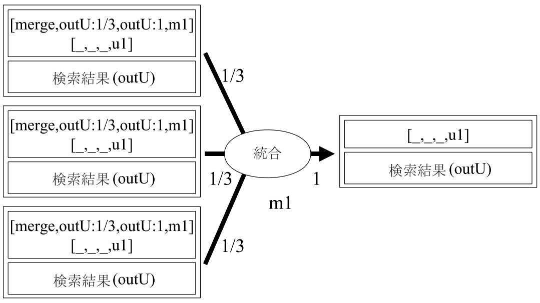

図 11　メッセージの統合

統合の後のふるまいを記述するサービスリストは, メッセージを複製して全ての検索エージェントに送り出す (2) の時点で各検索エージェントに対するメッセージ全てにつけておくこともできるが, そのほかに, 検索を行なうエージェントに送るメッセージには統合後のサービスリストを含めずに, サービスリストは別のメッセージとして統合を行なうエージェントに直接送り, 各検索エージェントを回った回答と統合後のサービスリストとを統合エージェント上で全て統合して次のサービスを行なうという方法も考えられる.

この統合の仕組みはユーザエージェントにまかせることも可能であり, その場合には統合を行なうサービス merge はそのままに, 実施するエージェントを u1 とすればよい.

(7)　ユーザエージェントは返ってきた回答 (図 11右) をまとめてユーザに提示する.

この例のようなメッセージ経路を設定することで, 回答メッセージの変換処理をファシリテータとは別のエージェントに完全に委譲し, 回答メッセージがファシリテータを経由しないようにできている. その結果として, メッセージがファシリテータに集中しない状態でファシリテータの機能の分散を行なうことができる. さらに, 変換する必要のない問い合わせや回答については (3) や (5) の処理を経路から完全に取り除くこともでき, データベースの選択処理を行なった結果単一のエージェントしか検索をしないことになった場合には, (6) の merge の処理を除くこともできる. その結果メッセージのやりとりの数が少なくなり, レスポンスの高速化を図ることができる.

6. ま　と　め

本稿では, SAGE: 仮想カタログのメッセージのやり取りを見直し, メッセージ指向のアプローチの導入を検討した. このアプローチでは, メッセージを単一のやりとりではなくさまざまなエージェントをたどる間に状態遷移していく一連のやりとりの流れを表すものとし, そのやりとりを通じて受ける処理内容をメッセージ自体に (サービスリストとして) 記述することとした.

このメッセージを受ける各エージェントは, 受け取ったメッセージに自分の持つサービスを適用し, 次へ回すだけでよい. これは, 従来エージェントに保持していた状態に関する部分をメッセージに持たせているので, エージェントをメッセージの管理に関してステートレスにできる (状態を保持する必要がない) ことによる.

また, サービスリストは解釈し書き換えることができるものとし, その性質を, エージェント系の実施すべきサービスをあらかじめ抽象的に記述しておいて, それを詳細化していく目的に使うことを検討した.

このメッセージ指向の枠組みの導入により, 効率の面からは, ファシリテータへの負荷の集中を回避し, かつ変換の必要のないメッセージについては直接データベースエージェントからユーザエージェントに結果を返すなどの経路設定が可能になり, 応答性を高めたり, 特定のエージェントに負荷が片寄らない構成をとることができることを示した. これは, メッセージ指向の導入により主にファシリテータをステートレスにできることによる. また, 設計や管理の面においてもメッセージの状態遷移に基づくアプローチが特に知識の分

散した状態において有効であることを述べた.

今後は実際にメッセージ指向の枠組みを実装し,SAGE の新しいアプリケーションに組み込んで実験・評価していきたい. その際には, 各エージェントの行なうサービスに対して抽象的な定義を与える必要がある. 現在, SAGE のファシリテータを拡張しつつ, 内部のサービスに定義を与える作業を行なっている段階であるが, これについても機会があれば報告したい.

また, このメッセージ指向のアプローチは複数のサービスにまたがって処理するようなアプリケーションとの適合性が非常に高い. その利点を活かし, 高速なレスポンスを必要とする分野だけでなく, 複雑なメッセージ配送を必要とする応用も積極的に検討していきたい.

文　献

[1] 菅坂玉美他, " 知的エージェント環境 SAGE の企業間 EC への応用 ", ソサイエティ大会併設特集シンポジウム,1997.

[2] 丸山文宏他, "SAGE: 仮想カタログ ", 情報処理学会第 54 回全国大会, 1997.

[3] 佐藤　陽他, "SAGE: 仮想カタログ – データベースのエージェント化 –", 情報処理学会第 54 回全国大会, 1997.

[4] M.R.Genesereth and S.P.Ketchpel, "Software Agents," Comm.ACM Vol.37 No.7, 1994.

[5] Michael R.Genesereth and Richard E.Fikes, "Knowledge Interchange Format Version 3.0 Reference Manual", June 1992.

[6] The DARPA Knowledge Sharing Initiative External Interfaces Working Group, "Specification of the KQML Agent-Communication Language", February 9 1994.