フレームマッピングによる準構造化情報の開放的統合エージェント*

伊藤 史朗　　上田 隆也　　池田 裕治
キヤノン (株) 情報メディア研究所
{itoh,takaya,ike}@cis.canon.co.jp

1997 年 12 月 19 日

Abstract

It is hard for information agents to integrate heterogeneous semi-structured information, since they use different structures to express the same information. A common ontology or ontologies could resolve this problem. However, it would be very difficult to build such an ontology for a dynamic information space such as the WWW. Thus, automatic reconciliation of differences is needed for open integrative agents. Example-Based Frame Mapping (EBFM) is the technique developed by us for this purpose. EBFM generates a frame map which has equivalent slot pairs of frame structure of the information. However, it is difficult to have a global map in multi-agent system handling a large information space. We propose a distributed EBFM where an agent has a local map and searches other maps to make correspondence with a frame out of his map if needed.

1　はじめに

インターネット上で発信される情報の増大に伴い，情報の検索，選択，分類，抽出，組織化等を行なう情報エージェントが数多く提案されている．多くの情報エージェントは，自らが対象とする情報のみを扱う．しかし，複数の情報エージェント間の協調動作や複数の情報源の利用には，複数の情報の関係を把握した上での情報統合が必要となる．

基本的な情報統合の一つに，同種ではあるが表現が異なる準構造化情報の統合がある．準構造化情報は，値に対する制約が緩い属性を定義し，属性を表現する項目を持つ文書として表現される情報であり [Malone87, Huffman97]，一つの情報源の中では同種の情報は同じ構造で表現される．インターネットでの代表的な情報発信手段である WWW では，メタタグにより明示的に，あるいはリストや表などの構造により暗黙のうちに準構造化情報が提供されることが多い．また，XML[Bray97] や RDF[Lassila97] といった構造化情報の記述形式の導入は，準構造化情報の明示的な表現をより増大させると考えられる．複数の準構造化情報の属性間の関係が分かれば，例えば欠落した情報を別の情報源から補完するといった処理が可能になる．このように，準構造化情報の統合エージェントは，インターネット上に分散する情報の利用に重要な役割を果たす．

情報統合エージェントが対象とする準構造化情報間の関係が全て定義されていれば，情報の統合は容易である．しかし，全ての関係が把握できるのは閉鎖系であって，WWW の持つ開放的な特徴と相反する．実際，情報発信者の視点の相違から，概念的には同じ情報であっても，異なる準構造化情報として発信されることがほとんどである．従って，準構造化情報の統合エージェントでは，異なる構造間の関係を獲得しながら準構造化情報を統合できねばならない．

従来，情報エージェントやマルチデータベース等の研究において，この問題は扱われてきた．マルチデータベースでは，グローバルなスキーマを作成する方法が考えられてきたが，この方法では大規模な情報空間を扱うにはコストがかかりすぎる [Goh94]．情報エージェントでは，対象とするドメインの知識やモデルをオントロジーとして作成し，これらから関係を導き出す方法が考えられてきた．一つのオントロジーを構築するのが難しい場合は，複数のオントロジーを統合して解決する方法も考えられている [Geddis95, Hammer95, 鷹合 96, 坂田 97]．しかし，WWW のような開放系では常に情報が変化するので，オントロジーの構築や統合を人手で行なっていては追従しきれない．そこで我々は，開放系においても自律的に統合を行なえる準構造化情報の開放的統合エージェントを提案する．

情報の開放的統合は，マルチエージェント間通信でも問題となる．KQML[Finin97] のようなエージェント間通信言語を用いる場合，通信の内容についてはコミュ

*Open Integrative Agents for Semi-structured Information by way of Flame Mapping.

Fumiaki Itoh, Takaya Ueda, and Yuji Ikeda.
Media Technology Laboratory, Canon Inc., 53, Imaikami-cho, Nakahara-ku, Kawasaki-shi, Kanagawa 211-8501, Japan.

第 6 回マルチ・エージェントと協調計算ワークショップ，日本ソフトウェア科学会，神戸市．

ニケーションオントロジー [溝口 97] の合意を前提としている．しかし，開放系において通信相手と常にオントロジーについての合意が得られる保証はない．それでも通信を実現するためには，異なるオントロジーの統合をエージェントが自律的に行なえる必要がある．情報エージェントにおいては，主要な通信は扱う情報の交換である．扱う情報が準構造化情報であれば，各エージェントが開放的統合機能を持てば，情報交換に関して自律的な通信が可能になる．

準構造化情報の開放的統合エージェントの実現には，少なくとも異なる構造定義における同一概念の自動対応付けが不可欠である．これには，我々が考案した「事例に基づくフレームマッピング (EBFM)」[伊藤 97] を利用する．EBFM は，異なる準構造化情報において同一概念を表わす属性を自動的に対応付ける手法である．EBFM は，この対応関係を一つのマップとして作りあげるので，サーチエンジンのような集中型のシステムには適しているが，複数のエージェント間の通信には向いていない．そこで本稿では，それぞれが部分的に構築したマップを持つエージェント同士が，マップを交換しながら，必要な対応関係を獲得する分散型の EBFM を提案する．

以下，2 章では準構造化情報をフレームとみなして用語を定義する．3 章では，フレームマップを自律的に作成しながら情報を交換できる開放的統合エージェントを提案し，そのための課題を指摘する．4 章では，フレームマッピングの手法である EBFM について説明する．5 章では，マルチエージェント間通信に必要な分散型の EBFM を提案する．6 章では関連研究との比較を行ない，7 章で今後の課題を述べる．

2 準構造化情報におけるフレーム

2.1 準構造化情報

準構造化情報は，弱く型付けされた項目に分かれた文書で表現される情報である．多くの場合，準構造化情報は一つのオブジェクトに関する情報であり，オブジェクトの特徴を示す属性を各項目で表現している．図 1は，プリンタ製品についての準構造化情報の例である．タグ付けされていないテキストでは，この例のように項目名と項目値を列挙する形で準構造化情報は表現されることが多い．項目値は属性の値を表現する文字列である．項目名は，その項目が表現する属性が何であるかを示す文字列である．情報の使い手が人間であり，項目値から属性を推定できる場合には省略されることもある．

図 2は，同じ情報を WWW で使われている HTML

品名 :LBP320
メーカー :キヤノン
解像度 (横) :2,400dpi
解像度 (縦) :600dpi
プリント速度 :6 枚 / 分

図 1: 準構造化情報の例

における表で表現した例である．表の各行が項目に相当し，1 列目で項目名を，2 列目で項目値を表現している．他にも HTML のリストで表現する方法などがあるが，いずれにしてもこれらのタグは表現構造を示すものであり，準構造化情報は暗黙のうちに表現されているにすぎない．

```
<TABLE>
<TR><TH>品名</TH><TH>LBP320</TH></TR>
<TR><TH>メーカー</TH><TH>キヤノン</TH></TR>
<TR><TH>解像度 (横)</TH><TH>2,400dpi</TH></TR>
<TR><TH>解像度 (縦)</TH><TH>600dpi</TH></TR>
<TR><TH>プリント速度</TH><TH>6 枚 / 分</TH></TR>
</TABLE>
```

図 2: HTML で表現した準構造化情報の例

そこで，HTML 中の準構造化情報を明示的に扱う方法として，メタタグが利用されている．図 3は，上記の例をメタタグで表現した例である．メタタグでは，準構造化情報の項目が明示的に表現されるので，項目の抽出が容易になる．しかしメタタグには，適用範囲を明示できない点や同一の文書内で複数のオブジェクトを表現する場合に問題がある．

```
<META NAME="品名" CONTENT="LBP320">
<META NAME="メーカー" CONTENT="キヤノン">
<META NAME="解像度 (横)" CONTENT="2,400dpi">
<META NAME="解像度 (縦)" CONTENT="600dpi">
<META NAME="プリント速度" CONTENT="6 枚 / 分">
```

図 3: メタタグで表現した準構造化情報の例

現在仕様を策定中の XML を用いると，上記の問題は解決される．図 4は，同じ例を XML で表現した例である．

ところで，一つの情報源において，同種のオブジェクトは，同じ構造の準構造化情報で表現されることが多い．同種のオブジェクトの集合をクラスと考えると，一つの情報源においては，同一クラスに属する情報は同じ構造の準構造化情報で表現されると仮定しても構わない．ところが，HTML では，メタタグを使ったとしてもクラスを明示的に記述できない．同一サイトにおいて同一のタグ構造 (表示構造) を持つ情報が同一

```
<PRINTER>
<MODEL>LBP320</MODEL>
<MANUFACTURER>キヤノン"</MANUFACTURER>
<HRESOLUTION>2,400dpi"</HRESOLUTION>
<VRESOLUTION>600dpi"</VRESOLUTION>
<SPEED>6 枚 / 分</SPEED>
</PRINTER>
```

図 4: XML で表現した準構造化情報の例

クラスのオブジェクトを表現していると推定可能だが，完全性，健全性とも保証はされない．しかし，XML や RDF を用いると，クラスを明示的に記述できる．

2.2 準構造化情報におけるフレーム

準構造化情報は，以下のように限定されたフレームで表現された知識と考えることができる．準構造化情報を交換する情報エージェントは，この限定されたフレームを内容とする通信を行なうことになる．

ここで扱うフレームでは，スロットは全てオブジェクトの属性を表現する．スロット値は，そのスロットが表現する属性の概念値 (属性値と呼ぶ) を表現する文字列だけをとる．スロット名は，そのスロットが表現する属性を明示的に表現する文字列である．スロット名，スロット値とも空文字列でも構わない．スロット s のスロット名を $name(s)$ で表わす．また，フレーム f で使用するスロット s の集合を $S_f(f)$ で表す．

あるオブジェクトについてフレームで表現された情報をインスタンスと呼ぶ．同一クラスに属するオブジェクトを同一フレームで表現したインスタンスの集合をソースと呼ぶ．ソース r における全インスタンスの集合を $Z(r)$ で表す．また，インスタンス z でのスロット s のスロット値を $val(z, s)$ で表す．ソース r で使用されるフレームを $f(r)$ で表す[1]．WWW の情報源はソースであり，情報エージェントの多くもソースを持つ．

オブジェクトとインスタンス，属性とスロット，属性値とスロット値は，それぞれ前者が概念上の実体であり，後者が表現上の実体である．インスタンス z が表現するオブジェクトを $obj(z)$ で，スロット s が表現する属性を $atr(s)$ で，スロット値 v が表現する属性値を $atrv(v)$ で表す．

[1] 以下の議論では，簡単のため異なるソースではフレームも異なると仮定する．同じフレームを使用する情報源が複数ある場合は，重複するオブジェクトを一つにまとめて新しいソースを作ることができるので，この仮定は問題ない．そこで，フレーム f で表現されるソースを $r(f)$ で表わす．また，ソースとフレームは 1 対 1 に対応するので，$S_f(f)$ を $S(f)$，$S_f(f(r))$ を $S(r)$，$Z(r(f))$ を $Z(f)$ と略記する．

2.3 フレームマップ

同じクラスに属するオブジェクトであっても，異なる情報源では異なるフレームで表現されることがある[2]．図 5は，先ほどのプリンタのオブジェクトが，異なる二つのフレームで表現されている例である．

品名	LBP320
メーカー	キヤノン
解像度 (横)	2,400dpi
解像度 (縦)	600dpi
プリント速度	6 枚 / 分

(a) 上記の準構造化情報のフレーム

	Canon
機種	LBP320
解像度	2,400dpi × 600dpi
用紙サイズ	A4

(b) 別のフレーム

図 5: 異なるフレーム表現の例

同一クラスの異なるソース r と r' 中には，同一のオブジェクトを表現しているインスタンスの対 z が存在する可能性がある[3]．そこで，$z \in Z(r), z' \in Z(r'), obj(z) = obj(z')$ である z を対応インスタンス対と呼ぶ．

また，各ソースで使用されているフレーム f と f' には，同じ属性を表わすスロットが，多くの場合存在する．同一クラスのオブジェクトであれば，情報の表現が異なっても，同一の属性に着目して表現する可能性は高いからである．そこで，$s \in S(f), s' \in S(f'), atr(s) = atr(s')$ である s を対応スロット対と呼ぶ．そして，フレームの集合 F について，F 中の任意のフレーム対について対応スロット対を保持したデータをフレームマップと呼ぶ．

3 フレームマッピング

3.1 情報統合エージェント

フレームマップが存在すると，扱うフレームが異なっても，次のような情報統合エージェントを実現できる．

- 情報仲介エージェント
  複数の情報源からの情報をまとめて仲介する．[Nishida93, Geddis95]
- 欠落情報補完エージェント
  ある情報源では欠落している属性情報を他の情報源から補完する．[Arens93]

[2] 言い換えれば，異なるソースが存在する．

[3] 以下，太字の記号で対を表わす．例えば，z はインスタンス対 (z,z') を表わし，s はスロット対を，f はフレーム対を表わす．

- 協調的情報推奨エージェント
  複数人の評価に基づいて情報を推奨するシステム [Resnick97] で，エージェントが評価も与える．

ここで利用する同一クラスの属性間の関係だけでなく，クラス間の関係や異なるクラス間での属性の関係を使った情報統合もある．例えば，プリンタクラスは製品クラスのサブクラスであるという関係や，メーカーを表わす属性値は企業クラスのオブジェクトになるといった関係を使うと，より複雑な処理が可能である．しかし，情報検索をタスクとした場合は，上記のエージェントが主要な情報統合エージェントである．

フレームマップが保持するのは，1 対 1 関係の対応スロット対だけであり，1 対多あるいは多対多の対応関係は利用できない．図 5の例において，(a) の 「解像度 (横)」と 「解像度 (縦)」の二つのスロットは，(b) の 「解像度」のスロットと対応しているが，この対応関係は利用できない．しかし，現実の情報では 1 対 1 の対応関係が 90% 以上の割合を占めているという調査結果 [伊藤 97] から，対応スロット対だけでも相当な処理を期待できる．

3.2 開放的統合エージェントとフレームマップピング

従来の情報統合エージェントでは，人手で作成したフレームマップを利用していた．これはエージェントの設計者が，エージェントが扱う情報源の範囲を限定できる閉鎖系でのみ可能な処理である．WWW のような開放系では常に情報が変化するので，フレームマップを人手で作成していては，エージェントは自律的に動作できない．

オントロジーについて合意した上で，エージェントを作成すれば動作できるという主張もある．しかし，情報エージェントでは，情報自体のオントロジーは相当大きいものであり，全てにおいて合意をとることは現実的ではない．また，情報発信方法をトップダウンに規定することなく各自が自由に発信できたことが，WWW で爆発的に情報が発信された理由の一つである．オントロジーの合意に従う制約は，この自由度を失わせる．

以上の点から，開放系においても自律的に情報の統合を行なえる開放的統合エージェントが必要である．その実現には，フレームマップを自動的に作成するフレームマッピング技術が欠かせない．

3.3 フレームマッピングの課題

フレームマッピングには，いくつかの課題がある．まず，同一の属性であっても表現が異なる場合が問題となる．これには，スロット名の相違とスロット値の相違の 2 種類があり，この二つが同時に起きることも多い．

以下のスロット対 s でスロット名の相違が起きている．

$$s \in S(r), s' \in S(r'), atr(s) = atr(s'), \\ name(s) \neq name(s')$$

図 5の例では，(a) の 「品名」と (b) の 「機種」が異なっている．また，(a) の 「メーカー」スロットに対応する (b) のスロットではスロット名が空文字列である．現実のデータでの調査によれば，スロット名の相違は対応スロット対の 80.4% で起きている．

スロット対 s で，以下のインスタンス対 z が存在するとスロット値の相違が起きる．

$$s \in S(r), s' \in S(r'), obj(z) = obj(z'), \\ atr(s) = atr(s'), atrv(val(z,s)) = \\ atrv(val(z',s')), val(z,s) \neq val(z',s')$$

図 5の例では，「メーカー」属性を表すスロット値が，「キヤノン」と 「Canon」になっている．同様の調査で，対応スロット対の 88.9% において，少なくとも一つの対応インスタンス対でスロット値の相違が起きている．

別の問題点は，スロット名が同じであっても，同じ属性を表現するスロットであるとは限らない点である．すなわち，以下のスロット対 s が存在する．

$$s \in S(r), s' \in S(r'), name(s) = name(s'), \\ atr(s) \neq atr(s'),$$

例えば，住所録情報において，スロット名が 「電話番号」であっても，あるフレームでは自宅の電話番号を表わし，別のフレームでは勤務先の電話番号を表わしているかもしれない．

また，フレームマッピングにスロット値の型を用いる方法が考えられるが，この場合も同様な問題が生じる．例えば電話番号を表わす文字列には特徴があるので，スロット値が 「電話番号」型であることは推定可能である．しかし，「電話番号」型を持つスロットであっても，あるフレームでは電話番号を表わし，別のフレームでは Fax 番号を表わしているかもしれない．

このように，スロット名やスロット値を単純に比較するだけでは，フレームマッピングは実現できない．

4 事例に基づくフレームマッピング

4.1 概要

前章で述べた課題を解決するために，我々が考えたのは，対応インスタンス対を求め，対応インスタンス対でのスロット値の対応関係から，対応スロット対

を導き出す方法である．以下，これを実現する事例に基づくフレームマッピング (Example-Based Frame Mapping)[伊藤 97] について簡単に説明する．

EBFM では，フレーム対 f における任意のスロット対 s が，対応スロット対である可能性を示す尺度 $rs(\mathrm{s})$ が得られる．そこで，フレームマップとしては，0 から 1 の実数値をとる尺度 $rs(\mathrm{s})$ を保持する．フレームマップを利用する場合は，$rs(\mathrm{s})$ が閾値 t を越えるスロット対 s が対応スロット対であると考える．図 6に，模式的に表わしたフレームマップの例を示す．

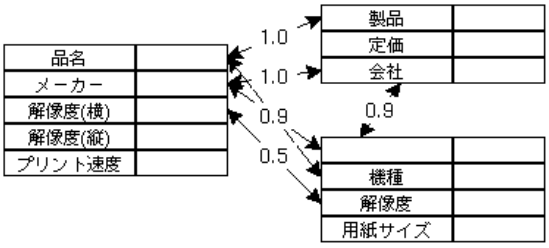

図 6: EBFM におけるフレームマップ

EBFM は，次の二つのフェーズから成る．

1. 事例ベースマッピングフェーズ
   フレーム対 f において，対応インスタンス対を事例として $rs(\mathrm{s})$ を求める．
2. 補完マッピングフェーズ
   上記のような事例がないフレーム対 f において，他のフレームを介して $rs(\mathrm{s})$ を求める．

4.2 事例ベースマッピングフェーズ

事例ベースマッピングフェーズでは，対応インスタンス対が存在するフレーム間で，対応スロット対を求める．しかし，対応インスタンス対は，そもそも対応スロット対が不明では得られない．そこで，任意のスロット対の集合 S が対応スロット対であるという仮説を生成し，仮説 S のもとで対応インスタンス対 z を求める．

4.2.1 仮説の信頼度

最終的に仮説を選択する戦略は後述するが，基本的には信頼度が高い仮説を選択する．仮説の信頼度 $rh(\mathbf{S})$ は，S により定まる対応インスタンス対の集合 Z から次式で求める．

$$rh(\mathbf{S}) = 1 - \prod_{\mathrm{z}\in\mathbf{Z}} (1 - rz(\mathrm{z}))$$

ここで，$rz(\mathrm{z})$ は，対応インスタンス対 z の信頼度であり，次式で求める．

$$rz(\mathrm{z}) = 1 - \prod_{\mathrm{s}\in\mathbf{S}} (1 - rzs(\mathrm{z},\mathrm{s}))$$

ここで $rzs(\mathrm{z},\mathrm{s})$ は，各スロット対 $\mathrm{s}(\in \mathbf{S})$ だけで z を同定する信頼度であり，次式で求める．

$$rzs(\mathrm{z},\mathrm{s}) = sim(v,v') \cdot sing(\mathrm{z},\mathrm{s}) \cdot rs'(\mathrm{s})$$

ここで，$v = val(z,s), v' = val(z',s')$ としている．

右辺の最初の sim は，s におけるスロット値の一致度である．最適照合検索 [Sato92] を利用して，文字列の一致度 (完全一致の場合に 1) を求め，この値を用いる．

次の $sing$ は，z のスロット値の組 (v,v') が，ソースの中で単独に存在する尺度である．z の他にもスロット値が一致するインスタンスがあるときは，z が同じオブジェクトを表わしている可能性は低い．$sing(\mathrm{z},\mathrm{s})$ は次式で求める．

$$sing(\mathrm{z},\mathrm{s}) = \frac{2sim(v,v')}{simall(v,r',s') + simall(v',r,s)}$$

ここで，

$$simall(v,r,s) = \sum_{z\in Z(r)} sim(v, val(z,s))$$

である．

最後の $rs'(\mathrm{s})$ は，フレーム対 s の信頼度である．フレームマップで，$rs(\mathrm{s})$ が得られている場合はその値を用いる．得られていない場合は，定数 C に，スロット対を構成する各スロットのユニーク度 $uniq(\{s\}, Z(r))$ を乗じた下記の値 rss を初期値として用いる．

$$rss = C \cdot uniq(\{s\}, Z(r)) \cdot uniq(\{s'\}, Z(r'))$$

ユニーク度は，あるインスタンス集合 $\hat{Z}$ において，スロット集合 $\hat{S}$ のスロット値がユニークかどうかを示す尺度である．ユニーク度は，情報理論に基づき以下のように定義される．

$$uniq(\hat{S},\hat{Z}) = \frac{1}{n(\hat{Z})\log n(\hat{Z})} \sum_{j=1}^{n(V)} \log \frac{n(\hat{Z})}{c_j}$$

ここで，集合 V は，$\hat{Z}$ 中の各インスタンスにおいて，$\hat{S}$ 中のスロット値を連結して得られる値の集合であり，c_j は V の j 番目の要素と同じ値をとる $\hat{Z}$ 中のインスタンスの個数を示す．また，集合 A の要素数を $n(A)$ で表している．ユニーク度は，各インスタンスが生起する事象 X と各値が生起する事象 Y との相互情報量 $I(X;Y)$

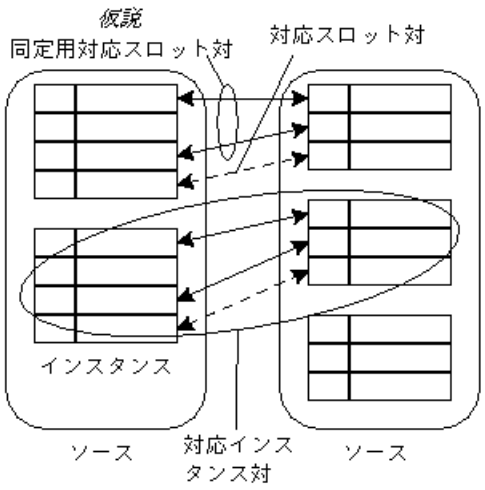

図 7: EBFM の概要

を事象 Xの平均情報量で割ったものである．ここで，相互情報量 $I(X;Y)$ は次式の通りである．

$$I(X;Y) = \frac{1}{n(\hat{Z})} \sum_{j=1}^{n(V)} \log \frac{n(\hat{Z})}{c_j}$$

4.2.2 同定用のスロットの分離

EBFM はスロット値だけを見て動作するので，スロット値が同じでも属性が異なるスロットは要注意である．例えば，定休日と特売日といったスロットは同じ値を取り得る．これらの対を仮説に含めると，対応インスタンス対の同定を誤る可能性がある．そこで，仮説に採用するスロットを，ユニーク度が高いスロットに限定する．図 7に，(同定用) 対応インスタンス対，対応スロット対，仮説の関係を示す．

仮説に含まれないスロットは，最終的に一つの仮説が採用されてから，求められた対応インスタンス対集合を用いて対応関係を求める．同定用スロット対を区別するため，フレームマップでは，同定用としての対応スロット対の信頼度 $ro(\mathrm{s})$ も保持する．

4.2.3 仮説選択の戦略

最終的には，仮説を構成する対応スロット対の個数ごとに信頼度が高い仮説を選び，求まる対応インスタンス対の個数が大幅に減少しない中で，対応スロット対数が最大となる仮説を選択する．これは，誤った仮説においても，一致度が高いスロット値の組が存在しえる問題を防ぐためである．

2 つ以上の対応スロット対から構成される仮説では，誤った対応付けが同時に起きる可能性は低いので，前述の戦略により誤った仮説は排除される．問題は，1 つの対応スロット対から構成される仮説である．この場合は，対応スロット対を構成する両スロットが，他のフレーム対に対するマッピングで使用されたことがある場合のみ選択する．

4.3 補完マッピングフェーズ

補完マッピングフェースでは，対応インスタンス対を持たないフレーム対 f に対するマッピングを行なう．既に対応関係が求められた他のフレームを介して，こうしたフレーム対の対応関係を求める．今，$s \in S(f), s' \in S(f'), s'' \in S(f'')$ であるスロットにおいて，s と s'が対応し，s'と s''が対応するなら，s と s''も対応するからである．補完マッピングフェーズでは，こうした対応スロット対を求める．

4.4 EBFM のアルゴリズム

以上の考えに基づいた EBFM のアルゴリズムは示す．

ステップ 1 フレームマップ Mが変化しなくなるまで，ステップ *2* 以降を繰り返す．

ステップ 2 同じクラス中の任意のフレーム対 f について，ステップ *3* 以降の処理を行なう．

ステップ 3 フレーム対 fの一方のフレーム fに対し，$S = \{s | s \in S(f), uniq(\{s\}, Z(f)) > t_u\}$ を求める．もう一方のフレーム f'についても同様に S'を求める．

ステップ 4 任意の仮説 $\mathbf{S} = \{\mathrm{s} | s \in S, s' \in S'\}$ を生成し，仮説を集めた仮説集合 Hを作成する．

ステップ 5 以下の条件を満足する仮説 S を Hから除く．ここで，$S(\mathbf{S}, r)$ は，S のうちソース r側のスロットだけを集めた集合を表わす．

- $uniq(S(\mathbf{S}, r), Z(r)) < 1$
- $uniq(S(\mathbf{S}, r'), Z(r')) < 1$
- $(n(\mathbf{S}) = 1) \wedge$
  $(\forall \bar{s}\, (ro(s, \bar{s}) \leq t_s) \vee (ro(s', \bar{s}) \leq t_s))$

ステップ 6 $\forall \mathbf{S} \in H$に対し，インスタンス対の集合 $\mathbf{Z} = \{\mathrm{z} | \forall \mathrm{s} \in \mathbf{S}\ sim(val(z, s), val(z's')) > t_c\}$ を求め，$rz(\mathrm{z})$ が降順となるように並べる．

ステップ 7 Z の任意の要素 z_kが次のいずれかの条件を満足する場合は，z_kを Z から除く．

- $\forall j\, 1 \leq j < k\, (z_j = z_k \vee z'_j = z'_k)$
- $\forall j\, k < j \leq n(\mathbf{Z})$
  $((z_j = z_k \vee z'_j = z'_k) \wedge rz(\mathbf{z}_k)/rz(\mathbf{z}_j) < t_z)$

ステップ 8 Hの中で $n(\mathbf{S})$ が等しい仮説ごとに，$rh(\mathbf{S})$ が最大である仮説をとり，それらを $n(\mathbf{S})$ の昇順で Kに並べる．

ステップ 9 Kの j番目の仮説 $\mathbf{S}_j$において求められたインスタンス対の数を n_jとした時，$n_{j-1}/n_j < t_h$を満足する最大の jをとる．そのような仮説がなければ $j = 1$ とする．

ステップ 10 採用された仮説 $\mathbf{S}$ において，$\mathrm{s} \in \mathbf{S}$ に対し，$rh(\mathbf{S}) > t_s$かつ $rh(\mathbf{S}) - ro(\mathrm{s}) > \delta$であれば，$ro(\mathrm{s}) = rh(\mathbf{S}))$ とする（$rs(\mathbf{S})$ についても同様）．ここで，δは正の微小量である．

ステップ 11 $\bar{S} = S(r) - S(\mathbf{S}, r)$, $\bar{S}' = S(r') - S(\mathbf{S}, r')$ としたとき，$L = \{\mathrm{s} | s \in \bar{S}, s' \in \bar{S}'\}$ を作成する．L を，

$$si(\mathrm{s}) = \frac{1}{n(\mathbf{Z})} \sum_{\mathbf{z} \in \mathbf{Z}} sim(val(z, s), val(z', s'))$$

の降順になるよう並べる．

ステップ 12 L から，次のいずれかの条件を満足する要素 s_kを除く．ここで，$Z(\mathbf{Z}, r)$ は，$\mathbf{Z}$ のうちソース r側のインスタンスだけを取り出した集合を表わし，$I_m(s, \bar{Z})$ は，$\hat{S} = \{s\}, \hat{Z} = \bar{Z}$としたときの先に定義した相互情報量を表す．

- $I_m(s_k, Z(\mathbf{Z}, r)) < t_m$
- $I_m(s'_k, Z(\mathbf{Z}, r')) < t_m$
- $\forall j\ 1 \leq j < k,\ s_j = s_k \vee s'_j = s'_k$

ステップ 13 $\mathrm{s} \in L$ に対し，$si(\mathrm{s}) > t_o$かつ $si(\mathrm{s}) - rs(\mathrm{s}) > \delta$であれば，$rs(\mathrm{s}) = si(\mathrm{s})$ とする．ここで，t_oは閾値である．

ステップ 14 以下の条件を満たす任意のスロットのリスト $L = (s_0, s_1, \ldots, s_n)$ を作成する．

- スロット s_jが属するフレームを $f_s(s_j)$ で表すとき，$\forall j \forall k\ 0 \leq j \leq n, 0 \leq k \leq n, j \neq k$ に対し，$f_s(s_j) \neq f_s(s_k)$．
- $n > 1$．

ステップ 15 $\mathrm{s}' = (s_0, s_n), \mathrm{s}_j = (s_{j-1}, s_j)$ と表したとき，$rsm = \prod_{j=1}^{n} rs(\mathrm{s}_j)$ なる rsm に対し，$rsm > t_s$かつ $rsm - rs(\mathrm{s}') > \delta$であれば，$rs(\mathrm{s}') = rsm$ とする．$ro(\mathrm{s})$ についても同様．

4.5 評価結果

WWW に実在する準構造化情報から，人手でフレームを抽出し EBFM の評価を行なった結果を示す．評価用に選んだ 9 クラスについて，それぞれ 10 個のソースから獲得した 20 個から 50 個の準構造化情報を使用した．クラスごとに 10 個のフレーム間でマッピングを行ない，人手で作成したフレームマップ (正解データ) と比較して適合率と再現率を求めた．以下に示すように適合率は健全性を，再現率は完全性を示す評価項目である．

$$\text{適合率} = \frac{\text{うち正しい対応スロット対数}}{EBFM\text{による対応スロット対数}}$$

$$\text{再現率} = \frac{\text{うち } EBFM\text{で得られた対応スロット対数}}{\text{正しい対応スロット対数}}$$

表 1に結果を示す．対応インスタンス対が n 個以上あるソース同士に限って求めた適合率と再現率も示している．n が 1 以上の場合は，補完マッピングフェーズだけで対応関係を求めたフレーム対は外れることになる．

表 1: EBFM の評価結果

	$\mathrm{n} = 5$	$\mathrm{n} = 3$	$\mathrm{n} = 1$	$\mathrm{n} = 0$
適合率	100.0%	99.4%	99.2%	98.7%
再現率	68.6%	65.1%	62.1%	31.1%

事例数が 5 以上あると適合率が 100%となっているが，事例が少ないと誤りが生じている．誤りが生じたのは，「音楽」クラスと「ホテル」クラスである．誤りは，価格や電話番号のスロットにおいて，最適照合検索では誤った値同士の一致度が高くなるケースで生じた．これに対し，数値の近さなど最適照合検索以外の一致度を併用することで，誤りを防げると考えている．

再現率は事例がある場合で 60%台である．これも一致度の問題であり，文字列として似ていないが属性値としては同じ値の一致度を求めていく方法を組み込んでいく必要がある．例えば，外来語におけるアルファベット表記とカタカナ表記の一致度は，カタカナ表記への変換 [住吉 94] を行なって一致度を求めればよい．事例がなく補完マッピングを行なう場合を含めると再現率は 30%である．評価にあたっては，ソースを任意に抽出したので，他との対応インスタンス対が全くないソースも含まれている．さらに多くのソースを用いれば，こうしたケースは減り，再現率は向上すると考えている．

5 分散型 EBFM

前章で説明した EBFM は，対象とするフレーム間の関係を一つのフレームマップに作りあげる．サーチエンジンのような集中型のシステムではこれで十分であるが，任意のエージェント間で情報交換を行なうようなマルチエージェントシステムでは，グローバルマップの構築は困難である．そこで本章では，各エージェントが独自のフレームマップを構築し，必要になった

時に他のエージェントのマップを参照しながら，対応関係を求める分散型の EBFM を提案する．

5.1 ローカルマップ

エージェント a_n が，フレーム集合 F_n 中のフレームを対象として EBFM により作ったフレームマップ m_n を，a_n のローカルマップと呼ぶ[4]．エージェント a_n は，F_n の要素を独自の戦略で決定する．それまでに遭遇したフレームを全て対象としてもよいが，大量の記憶領域を必要とするので，ほとんど関係することがないフレームは外すべきである．

5.2 探索による対応付け

エージェント a_o がエージェント a_d と情報を交換する時，f_d が F_o の要素でない場合，a_o のローカルマップだけではフレーム間の対応が得られない．そこで，他のエージェントを介して f_o と f_d の対応関係を得るための探索が必要になる．すなわち，ローカルマップ m_o 中で f_o との対応関係が得られている他のフレーム f_k を選び，a_k のローカルマップ m_k を用いて，f_k と f_d との対応関係が得られないかを調べる．得られない場合は，最終的に f_d との対応関係が得られるまで探索を行なう．

この探索は，フレームがノードとなり，ローカルマップ中の対応関係がリンクとなるノード o からノード d への探索木を構成する．フレーム f_x に対応するノード x は，データ $Np(x), Nc(x), E(x), B(x), lc(x)$ を持つ．最初の $Np(x)$ は，x の親ノードのリストである．この探索では，スロットごとに別のリンクを辿る可能性があるため，親ノードが 2 つ以上になることがある．次の $Nc(x)$ は，x の子ノードのリストである．次の $E(x)$ は，x と o 間で対応付けられた対応スロット対 s_e のデータリストである．$E(x)$ の要素 $e(x)$ は，データ組 $(so(x), ss(x), rs(x))$ である．s_e のうち f_o のスロットが $so(x)$，f_x のスロットが $ss(x)$，s_e のスコアが $rs(x)$ である．次の $B(x)$ は，ノード x から探索可能なノードに関するデータのリストである．$B(x)$ の要素 $b(x)$ は，探索可能なノード $nb(x)$ と，$nb(x)$ と o 間で対応付けられる対応スロット対のデータ $Eb(x)$ とから構成される．ここで，$Eb(x)$ の要素 $eb(x)$ は，$e(x)$ と同様，$(sob(x), ssb(x), rsb(x))$ である．最後の $lc(x)$ は，ノード x からノード d への探索時点で判明しているリンク数である．なお，探索されたノードの集合を N で表わす．また，$C = \{b_j(n) | \forall j\ b_j(n) \in B(n),\ \forall n \in N\}$ とする．

探索は C 中から適当な $b(k)$ を選び，ノード k からノード $nb(k)$ へとリンクを延ばして行なう．

5.3 探索戦略

この探索空間が大きいと現実的には探索しきれない．そこで探索戦略が重要になる．探索木を延ばす $b(k)$ を，以下の優先順位で選択する戦略をとる．

1. $nb(k) \in N$ であれば，$lc(nb(k))$ が小さいもの．
2. $Eb(k)$ の要素数が多いもの．
3. $h = nb(k)$ としたとき，m_h での対応スロット対数が多いもの．

1 は，既にノード d に到達可能と判明しているノードの優先を意味している．あるノードが d に到達可能と分かっていても探索を続けるのには次の二つの理由がある．一つは，ローカルマップ間に双方向性がない ($f_k \in F_h$ であっても $f_h \in F_k$ とは限らない) ので，双方向探索を行なうからである．反対方向から探索を行ないノード x に到達すれば，$lc(x)$ が更新される．もう一つは，f_o の一部のスロットは対応付けられたが，他のスロットは対応が付いていない場合に，探索を続ける必要があるからである．あるスロットでの対応関係を得たときの探索経路にあるノードは，当然 d に到達可能なノードである．これらのノードは，新たに対応関係を求めるときも，別のスロットの対応付けに寄与する可能性が高い．2 は，対応スロット対の数が多いフレーム間の優先を意味している．この方が，一度に対応関係を得られる可能性が高い．3 は，他のフレームへの対応関係を多く持つノードの優先を意味している．

ノード k から探索を行なう場合，ノード k の親ノード $np(k)$ のローカルマップにあるフレームへの探索は必要ない．これらの対応関係は，ローカルマップ $m_{np(k)}$ で既に求まっているからである．逆に，距離が 2 以上離れているノード間[5]では，EBFM を行なうと新しい対応関係が求まる可能性がある．ここで，EBFM を適用するのは，多くの事例を含むソース間が適当である．事例が多い方が，対応関係が求まる可能性が高いからである．問題は，探索過程のどこで EBFM を適用するかであり，これは今後の課題である．

5.4 探索アルゴリズム

以下，探索のアルゴリズムを説明する．この処理を a_o と a_d から双方向に行なう．

ステップ 1 以下のようにノード o を作成し，N_m に o を加える．$Np(o) = \{\}$,$Nc(o) = \{\}$, $E(o) =$

[4] 簡単のため，エージェントが使用するフレームは全て異なると仮定し，エージェント a_n 使用するフレームを f_n で表わす．F_n には当然 f_n が含まれる．また，f_n に対応するソースを r_n で表わす．

[5] 距離が無限大，すなわち到達していないノード間も含む．

$\{(s,s,1.0)|\forall s \in S(f_o)\}$, $lc(o) = \infty$. また，$\forall f_k \in F_o$ に対して，以下のように作成する $Eb(o)$ が空でなければ，$nb(o) = k$とした $b(o)$ を $B(o)$ に追加する．

m_oあるいは m_kにおいて，$rs(\mathrm{s}) > t(s \in S(f_o), s' \in S(f_k))$ であれば，$(s, s', rs(\mathrm{s}))$ を $Eb(o)$ の要素とする．

ステップ 2 以下のようにノード d を作成し，N_m に d を加える．$Np(d) = \{\}$,$Nc(d) = \{\}$,$E(d) = \{\}$,$B(d) = \{\}$, $lc(d) = 0$.

ステップ 3 Cが空の場合は終了．空でない場合は，前章の探索戦略に基づいて Cから $b_i(k)$ を選択する．

ステップ 4 $Nc(k)$ に $h(h = nb_i(k))$ を加え，$B(k)$ から $b_i(k)$ を削除する．$h \notin N$の場合はステップ *5* に，$h \in N \wedge h \notin N_m$の場合はステップ *6* に，$h \in N_m$の場合はステップ *8* に飛ぶ．

ステップ 5 以下のようにノード h を作成する．$Np(h) = Nc(h) = E(h) = B(h) = \{\}, lc(h) = \infty$.

ステップ 6 ノード h を次のように変更し，N_mに h を加える．$Np(h) = \{k\}$,$Nc(h) = \{\}$,$E(h) = Eb_i(k)$, $B(h) = branch(h, k, E(h))$.

ここで，$branch(h, k, E)$ は，$\forall f_l \in F_h$に対して，次の条件を満足する b*(*ただし $nb = l$*)* の集合である．*1)*$f_l \notin F_k$である．*2)* 以下の処理で作成する Eb が空でない．

m_hまたは m_lにおいて $rs(\mathrm{s}) > t(s \in S(f_h), s' \in S(f_l))$ であり，$\exists j\ ss_j = s$ かつ $rs_j \cdot rs(\mathrm{s}) > t$ であれば，$(so_j, s', rs_j \cdot rs(\mathrm{s}))$ を Eb の要素とする．

ステップ 7 ノード o から h への最短パスのリンク数 $lco(h)$ を，もう一方のエージェントへ通知し，ステップ *9* に飛ぶ．

ステップ 8 まず，$lc(k) > lc(h) + 1$ であれば $lc(k) = lc(h) + 1$ とする．次に，$Np(h) = Np(h) + \{k\}$ とする．さらに，更新処理 $update(h, k, Eb_i(k))$ を行なう．また，$\exists j\ nb_j(h) = k$であれば，$update(k, h, Eb_j(h))$ を行なう．

更新処理 $update(h, k, Eb)$ は以下の通りである．まず，空リスト $Et(h)$ を用意する．そして，$\forall i\ s_i = sob_i$に対して次の処理を行なう．まず，$\forall j\ ss_j(h) \neq s_i$であれば，$eb_i$を $E(h)$ と $Et(h)$ に追加する．また，$\exists j\ ss_j(h) = s_i$ かつ $rs_j(h) < rsb_i$であれば，$E(h)$ の要素 $e_j(h)$ を eb_i で置き換え，eb_iを $Et(h)$ に追加する．

次に，*Et(h)* が空でなければ以下の処理を行なう．$Bt(h) = branch(h, k, Et(h))$ を求め，$Bt(h)$ を $B(h)$ に追加する．$\forall j\ nbt_j(h) \in Np(h) \cup Nc(h)$ である jに対し，$update(nbt_j(h), h, Ebt(h))$ を行なう．

ステップ 9 $Et(d)$ が空でない場合は，以下の処理を行なう．$s = so_j(d)$ のとき，$\forall b(l) \in C$に対して，$sob(l) = s$ である要素を $Eb(l)$ から削除する．その結果 $Eb(l)$ が空になった場合は $b(l)$ を削除する．

ステップ 10 $lco(h)$ の通知があった場合に，$h \in N$であれば $lc(h) = lco(h)$ とする．$h \notin N$であれば，以下のようにノード h を作成する．$Np(h) = Nc(h) = E(h) = B(h) = \{\}, lc(h) = lco(h)$. そしてステップ *3* に戻る．

6 関連研究

本研究の目的は，開放的情報統合エージェントを実現するために必要な自動的フレームマッピング方法を提供することである．自動的なフレームマッピングを行なう方法はいくつか提案されているが，いずれもスロットの対応関係を得るヒューリスティクスを用いており，任意のスロットへの拡張に問題があるか，マッピングの精度に問題がある．

Huffman らは，スロット名と値の型に関するヒューリスティクスにより，スロットの役割を推定している [Huffman97] ．この手法を用いれば，別のフレームにおいて同じ役割を果たすスロット同士を対応付けることができる．しかし役割の候補は人手で準備されるので拡張性に問題がある．Semint[Li95] は，各スキーマおよびインスタンスの統計量を基礎にグローバルスキーマを導き出すマルチデータベースである．しかし，統計量はヒューリスティクスなルールに基づいて算出されており，一般性に欠ける．また精度も高くない．

自動フレームマッピングに近い研究として，鷹合らによるオントロジー統合の支援環境 Donden[鷹合 96] がある．スロット名の類似度によりクラスタ分析を行ないクラスの類似度を求めている．インスタンスの類似度については詳細な記述がないが，仮にスロット値から同様の方法によるとしても，同一概念を表現するスロットの対応付けを得ることはできない．Ambite らは，モデル (フレームに相当) とインスタンスの不整合を検出してモデルを修正する方法を提案している [Ambite95] ．この手法を用いて近似するフレーム同士の変換ができれば，元のフレームの対応関係を求められる．しかし，この方法ではインスタンスを一意に定めるキー属性の存在を仮定している．この仮定は一般的には通用しない．高橋らは，電話帳のような目録情報と WWW 上の一般情報を統合するために，目録情報と一般情報のフレームマッピングの利用を提案している [高橋 97] ．ここでの対応付けは電話番号スロットだけであり，汎用的に行なう手法は示されていない．

マルチデータベースシステム [Collet91, Arens93, Levy96] や他の情報統合エージェント [Farquhar95,

Geddis95, Hammer95, 坂田 97] では，グローバルスキーマやフレームマップの作成を人手で行なっている．そのため開放的な統合は困難である．

7 おわりに

WWW のような開放系でも準構造化情報の構造の違いを解消して，情報統合を行なえる開放的統合エージェントを提案した．異なる構造を対応付ける技術として，事例に基づくフレームマッピングを用いる．マルチエージェントシステムでは，エージェントが持つフレームマップがそれぞれ異なるので，マップ情報を交換して対応付けを行なう分散型のフレームマッピングを実現する探索手法を提案した．

今後の課題として，探索過程においても EBFM を適用する方法の検討がある．また，クラスが異なるフレーム間で探索を行なうと，全探索を行なわない限り対応しないことが判明しない．異なるクラスの情報を同時に扱う場合には，クラスの違いを識別しながら探索を行なう方法が欠かせない．

参考文献

[Ambite95] Ambite, J. L. and C. A. Knoblock: Reconciling Distributed Information Sources, in *AAAI Spring Symposium on Information Gathering from Heterogeneous Distributed Environments*, pp. 1–5, 1995.

[Arens93] Arens, Y., C. Y. Chee, C. Hsu, and C. A. Knoblock: Retrieving and Integrating Data from Multiple Information Sources, *International Journal of Intelligent and Cooperative Information Systems*, Vol. 2, No. 2, pp. 127–158, 1993.

[Bray97] Bray, T., J. Paoli, and C. M. McQueen: *Extensible Markup Language*, W3C Proposed Recommendation, http://www.w3.org/TR/PR-xml-971208, 1997.

[Collet91] Collet, C., M. N. Huhns, and W. M. Shen: Resource Integration Using a Large Knowledge Base in Carnot, *IEEE Computer*, Vol. 24, No. 12, pp. 55–11, 1991.

[Farquhar95] Farquhar, A., A. Dappert, R. Fikes, and W. Pratt: Integrating Information Sources Using Context Logic, in *AAAI Spring Symposium on Information Gathering from Heterogeneous Distributed Environments*, pp. 47–50, 1995.

[Finin97] Finin, T., Y. Labrou, and J. Mayfiled: KQML as an Agent Communication Language, in Bradshaw, J. ed., *Software Agents*, MIT Press, Cambridge, MA, 1997.

[Geddis95] Geddis, D. F., M. R. Genesereth, A. M. Keller, and N. P. Singh: Infomaster: A Virtual Information System, in *CIKM Workshop in Intelligent Information Agents*, 1995.

[Goh94] Goh, C. H., S. E. Madnick, and M. D. Siegel: Context Interchange: Overcoming the Challenges of Large-scale Interoperable Database Systems, in *Proc. of the 3rd International Conference on Information and Knowledge Management*, 1994.

[Hammer95] Hammer, J., H. Garcia-Molina, K. Ireland, Y. Papakonstantinou, J. Ullman, and J. Widom: Information Translation, Mediation, and Mosaic-Based Browsing in the TSIMMIS System, in *Proc. of ACM SIGMOD International Conference on Management of Data*, 1995.

[Huffman97] Huffman, S. B. and C. Baudin: Toward Structured Retrieval in Semi-structured Information Spaces, in *Proc. of the Fifteenth International Joint Conference on Artificial Intelligence*, pp. 751–756, 1997.

[伊藤 97] 伊藤, 上田, 池田 :分散情報源に対する情報エージェントのための事例に基づくフレームマッピング, ソフトウェアエージェントとその応用シンポジウム, pp. 78–86, 1997.

[Lassila97] Lassila, O. and R. R. Swick: *Resource Description Framework Model and Syntax*, W3C Working Draft, http://www.w3.org/TR/WD-rdf-syntax, 1997.

[Levy96] Levy, A. Y., A. Rajaraman, and J. J. Ordille: Query-Answering Algorithms for Information Agents, in *Proc. of the Thirteenth National Conference on Artificial Intelligence*, pp. 40–47, 1996.

[Li95] Li, W.: Knowledge Gathering and Matching in Heterogeneous Databases, in *AAAI Spring Symposium on Information Gathering from Heterogeneous Distributed Environments*, pp. 116–121, 1995.

[Malone87] Malone, T. W., K. R. Grant, K. Y. Lai, R. Rao, and D. Rosenblitt: Semistructured Messages Are Surprisingly Useful for Computer-Supported Coordination, *ACM Transactions on Office Information Systems*, Vol. 5, No. 2, pp. 115–131, 1987.

[溝口 97] 溝口, 池田 :オントロジー工学序説 —内容指向研究の基盤技術と理論の確立を目指して—, 人工知能学会誌, Vol. 12, No. 4, pp. 559–569, 1997.

[Nishida93] Nishida, T. and H. Takeda: Towards the Knowledgeable Community, in *Proc. Int. Conf. on Building and Sharing of Very-Large Scale Knowledge Bases '93*, pp. 157–166, 1993.

[Resnick97] Resnick, P. and H. R. Varian: Recommender Systems, *Comm. of the ACM*, Vol. 40, No. 3, pp. 56–58, 1997.

[坂田 97] 坂田, 多田, 大竹 :WWW 上における Metadata の記述とオンラインショッピングへの応用, 第 54 回情処全国大会論文集, 3, pp. 151–152, 1997.

[Sato92] Sato, S.: CTM: An Example-Based Translation Aid System, in *Proc. of the Fifteenth International Conference on Computational Linguistics*, pp. 1259–1263, 1992.

[住吉 94] 住吉, 相沢 :英語固有名詞の片カナ変換, 情処学論, Vol. 35, No. 1, pp. 35–45, 1994.

[鷹合 96] 鷹合, 武田, 西田 :マルチエージェントシステムのためのオントロジーの分散開発支援環境, 第 10 回人工知能学会全国大会論文集, pp. 139–142, 1996.

[高橋 97] 高橋, 三浦, 西部, 島 :不均一で分散した情報の構造化と統合–情報統合ディレクトリ–, ソフトウェアエージェントとその応用シンポジウム, pp. 93–98, 1997.