3G3-04

モバイルエージェントのための位置透過な操作手法とエージェント分散処理への応用

A location transparency method for mobile agents and application of agents' distributed system

山谷 孝史[*1] Takafumi YAMAYA　服部 宏充[*2] Hiromitsu HATTORI　伊藤 孝行[*1] Takayuki ITO　新谷 虎松[*1] Toramatsu SHINTANI

[*1]名古屋工業大学 大学院 情報工学専攻 Graduate School of Engineering, Nagoya Institute of Technology

[*2]日本学術振興会 JSPS Research Fellow

In this paper, we propose an implementation technique based on a location transparent agent operating method for a mobile agent framework *MiLog*. When we request an operation to an agent on another remote platform, we need to identify agent's location, e.g. IP address. However agent's location management is complicated. In this paper, we create Access Agents to realize a location transparency in mobile agent-based applications. An Access Agent always manages the agent's location. When we request an operation to an agent, we request to its Access Agent instead of requesting to the agent. In this method, we can build applications using mobile agents without considering agents' location.

1. はじめに

分散システムの柔軟な運用を実現する方法として，モバイルエージェントに基づくシステムの構築が考えられる．モバイルエージェントは，コンピュータ間の自律的移動能力を持つプログラムである [Fuggetta 98]．モバイルエージェントは，計算機間を移動しながら，継続的に処理を行うことが可能である．そのため，モバイルエージェントは，次世代の分散システムの中核技術として，近年注目を集めている．

モバイルエージェントに基づくシステムでは，ネットワーク上の他の計算機上で動作中のエージェントに処理を要求する際，エージェントの位置情報を指定しなければならない．そのため，開発者は，エージェントが存在する計算機の，IP アドレス等で示される論理的な位置情報を考慮してプログラムを行う必要がある．しかし，計算機間を移動するエージェントの位置情報の管理は煩雑である．モバイルエージェントの特長である計算機間の自律移動は，システムの開発と運用の双方を困難にしている．開発者の負担を軽減し，モバイルエージェントの特徴を活かしたシステムを構築するために，エージェントの移動や位置情報を意識することなく，システムを開発する仕組みが必要となる．本稿では，モバイルエージェントの位置情報を管理する負担を解消するために，位置透過性を持つモバイルエージェント操作機構を実現する．ここでの位置透過性とは，ユーザが資源の配置されている場所を，システムに指示することなく資源を利用できる性質を指す．位置透過性の実現により，エージェントの位置情報を意識することなく，エージェントに処理要求を送信できる．開発者は，動的に変化するエージェントの位置情報の管理が不要となり，開発効率の向上が期待できる．また，運用時のエージェント管理においても，他の計算機へ移動したエージェントに対するアクセスが容易化され，処理要求を効率化できる．

本稿の提案手法では，位置透過性の実現のために，移動したエージェントへのアクセスポイントとして機能するアクセスエージェントを用いる．アクセスエージェントは，エージェントが移動した際，移動元に作成され，移動したエージェントの位置を常時捕捉する．移動したエージェントとの通信は，移動先ではなく，アクセスエージェントへ問い合わせることで実現できる．さらに，本稿では，提案手法に基づいたアプリケーションとして，モバイルエージェントビューワ *MiSight* を実装する．*MiSight* を用いることで，開発者は，エージェントのネットワーク上での位置を透過的に扱い，エージェントの挙動確認，及びエージェントへの問い合わせを実行できる．

本稿の構成は以下の通りである．2 章では，モバイルエージェントに基づくシステム開発の問題点を述べる．3 章で，モバイルエージェントフレームワーク *MiLog*[Fukuta 00] における位置透過なエージェント操作手法を提案する．4 章では，3 章で述べた提案手法をより詳細に述べる．5 章で，本手法を用いたエージェントビューワ *MiSight* の実装について述べ，6 章で考察を述べる．最後に 7 章でまとめを述べる．

連絡先: 山谷孝史，名古屋工業大学大学院情報工学専攻新谷研究室，〒 466-8555 愛知県名古屋市昭和区御器所町，電話 052-735-7968，FAX 052-735-5477，yamaya@ics.nitech.ac.jp

2. モバイルエージェントに基づくシステム開発の問題点

モバイルエージェントシステムでは，エージェントは任意に移動するため，エージェントが位置するプラットフォームは動的に変化する．エージェントに対する問い合わせのためには，IP アドレスで示されるプラットフォームの位置情報を付加する必要がある．しかし，エージェントの位置が動的に変化するモバイルエージェントシステムでは，位置情報を厳密に管理するのは煩雑なタスクである．そのため，システム開発過程におけるテストデバッグは困難となる．分散システムのデバッグの困難さは知られているが [Kubota 98]，ここでは，エージェントの位置情報の動的変化への対応が必要となる．また，システムの運用時には，エージェント間の通信の信頼性が高いレベルで保証されなければならない．位置情報の管理の煩雑さを解消することによる，システム動作の確実性の向上が必要である．

3. *MiLog* における位置透過的な操作手法

本章では，モバイルエージェントフレームワーク *MiLog* における位置透過性の実現手法について述べる．本手法では，位置透過性を実現するために，アクセスエージェントを用いる．アクセスエージェントは，他のプラットフォームに移動したエージェント (以下，本稿ではオリジナルエージェントと呼ぶ)

の移動元のプラットフォームに，オリジナルエージェントと同じ名前で作成されるエージェントである．アクセスエージェント自体は，１つのモバイルエージェントとして実現できるので，本提案手法は *MiLog* に限らず，他のモバイルエージェント環境にも実装可能である．

アクセスエージェントは，オリジナルエージェントの移動が成功した後に作成されるアクセスエージェントには，オリジナルエージェントの持つ処理手続きは与えないため，オリジナルエージェントに代わって実処理は行わない．代わりに，アクセスエージェントへの問い合わせは，保持しているオリジナルエージェントの位置情報に基づき，オリジナルエージェントへ送信される．*MiLog* では，エージェントの問い合わせに専用の通信用述語を用いる．代表的な通信用述語に query がある [Hattori 03]．query は他のエージェントに対して処理を問い合わせ，処理の終了を待つ．例えば，同一プラットフォーム上のエージェント tester に対して問い合わせ operation を行う場合，以下のように記述する．

```
query(tester,operation).                    (1)
```

他のプラットフォーム上の tester に対して問い合わせを行う場合，プラットフォームの IP アドレスを付加し，以下のように記述する．

```
query(tester,operation,<IP アドレス>).      (2)
```

アクセスエージェントは，述語 (1) の形式で受信した問い合わせを，述語 (2) の形式へ変換，送信する．そのため，アクセスエージェントを用いることにより，本来述語 (2) の形式で記述しなければならない処理を，述語 (1) の形式で記述できる．また，アクセスエージェントとオリジナルエージェントの名前は同じであるため，名前を区別する必要はない．

問い合わせ形式の変換のために，アクセスエージェントは，オリジナルエージェントの位置情報を常時保持する．具体的には，アクセスエージェントは，文献 [Milojicic 98] の registering モデルにおけるネームサーバとして機能する．ここでは，オリジナルエージェントが移動を行った際に，位置情報を逐次更新する．文献 [Milojicic 98] のモデルでは，単一のネームサーバが全てのエージェントの位置情報を管理する一方，本手法では，各エージェント専用のアクセスエージェントが位置情報を管理する．アクセスエージェントは，オリジナルエージェントの移動毎に管理している位置情報を更新する．本手法では，位置情報の管理を複数のプラットフォームに点在する，複数のアクセスエージェントが行う．そのため，局所集中的なネットワークアクセスが生じにくい．また，文献 [Milojicic 98] のモデルでは，ネームサーバの障害がシステム全体に影響するのに対し，本手法では，特定のアクセスエージェントに障害が発生しても，システム全体への影響が低く抑えられる．本手法により，エージェント生成後は，エージェントの位置情報は不変であるとして処理を記述できる．

本手法の最大の利点は，*MiLog* に対して，最小限の変更を行うだけでエージェントの位置透過性を実現したことである．具体的には，エージェントの移動時にアクセスエージェントを生成する機構を構築するだけで，位置透過的なエージェント操作機構を実現している．また，アクセスエージェントとオリジナルエージェント間の処理は，*MiLog* におけるエージェント間通信の仕組みで実現でき，特別な機構の実装は必要ない．また，アクセスエージェント自体は，１つのモバイルエージェントとして実現できるので，本手法は *MiLog* に限らず，他のモバイルエージェント環境にも実装可能である．

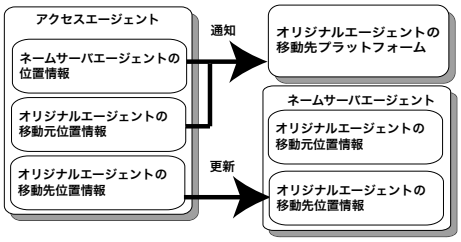

図 1: アクセスエージェントの作成時の処理

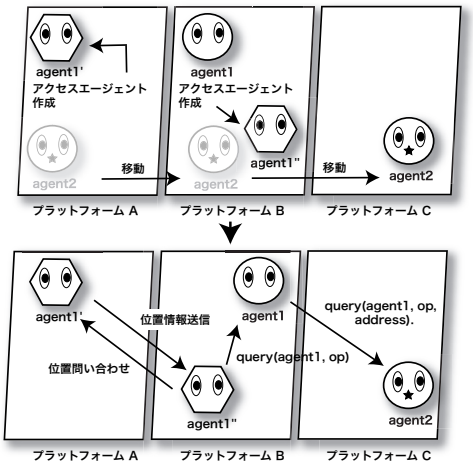

図 2: エージェントの複数回移動時の処理手順

4. 位置情報管理の詳細

4.1 複数回移動時の位置情報管理

本節では，オリジナルエージェントが複数回の移動を行った際の詳細な位置情報管理手法について述べる．

オリジナルエージェントの移動時，移動元のプラットフォームに，アクセスエージェントが作成される．アクセスエージェントは以下の情報を内部に保持する．

- ネームサーバエージェントの位置情報
- オリジナルエージェントの移動先プラットフォームの位置情報
- オリジナルエージェントの以前移動してきた元プラットフォームの位置情報

ネームサーバエージェント (以後，NS エージェントと表記する) とは，オリジナルエージェントの位置情報を集中的に管理するアクセスエージェントである．オリジナルエージェントが初回移動を行った際の移動元アクセスエージェントが NS エージェントとなる．図 1 はアクセスエージェント作成時の処理をまとめたものである．アクセスエージェントは，自身の生成時に NS エージェントの保持するオリジナルエージェントの位置情報を，オリジナルエージェントの移動先の位置情報で更新する．また，オリジナルエージェントの移動先であるプラットフォームに対して，NS エージェントの位置情報および自身の位置情報を通知する．通知された位置情報は，通知されたプラットフォームでアクセスエージェントが作成された際に保持される情報となる．アクセスエージェントはオリジナルエージェントの位置情報が必要になった際，NS エージェントに問い合わせることによって位置情報を取得する．

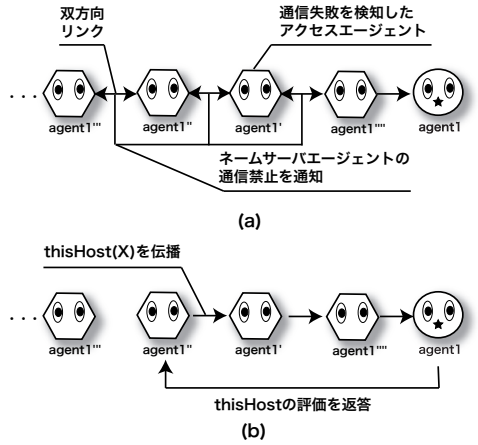

図 3: NS エージェントの再構築

また，図 2 は，エージェント移動時のアクセスエージェントの動作の流れを表している．本例では，agent1 がプラットフォーム A からプラットフォーム C までの移動を行い，プラットフォーム B 内で問い合わせが発生した時の処理を表している．便宜的に，agent1 に対する，プラットフォーム A 上のアクセスエージェントを agent1’，プラットフォーム B 上のアクセスエージェントを agent1” と記す．

(1) agent1 がプラットフォーム B へ移動．

(2) プラットフォーム A 上に agent1’ を作成．agent1’ は，NS エージェントとなる．

(3) agent1 がプラットフォーム C へ移動．

(4) プラットフォーム B 上に agent1” を作成．agent1’ の位置情報を保持する．

(5) プラットフォーム B 上で agent1” へ問い合わせを実行．

(6) agent1” が agent1’ から agent1 の位置情報を取得．

(7) agent1 へ問い合わせが送信され，処理される．

本手法により，アクセスエージェントを用いて位置透過的な操作が可能である．しかし，NS エージェントが消失した際，位置情報の取得ができない．そのため，NS エージェントを再構築する必要がある．

4.2 NS エージェントの動的再構築手法

アクセスエージェントはオリジナルエージェントの移動先位置情報および移動元位置情報を保持する．そのため，アクセスエージェント同士は移動先および移動元位置情報によって双方向にリンクを構成している．本手法では，アクセスエージェントのリンクを用いて，NS エージェントを再構築する．本手法の処理手順を以下の (1) から (3) に示す．

(1) アクセスエージェントに対して，NS エージェントへの問い合わせをブロックする．図 3 の (a) は，NS エージェントとの通信禁止通知を送信している．NS エージェントとの通信が失敗した場合，通信失敗を検知したアクセスエージェントは，リンクに対して，NS エージェントへの通信を禁止させる．

(2) オリジナルエージェントの位置情報を取得する．図 3 の (b) はオリジナルエージェントへ問い合わせが伝播する処理を表している．通信失敗を検知したアクセスエージェントは，リンクに対して，エージェントの位置情報を取得する述語を評価させる．具体的には，以下の述語を評価する．

```
thisHost(X).
```

リンクへの問い合わせは，最終的にオリジナルエージェントへ伝播し，評価結果は通知元のアクセスエージェントに返される．

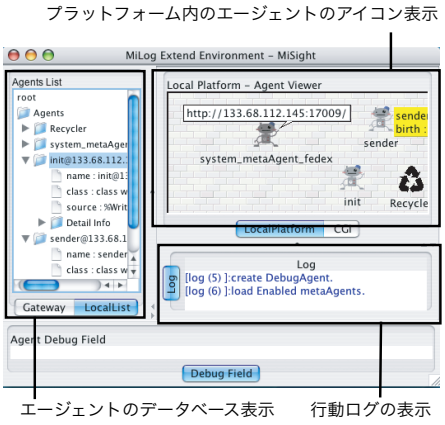

図 4: *MiSight* によるエージェント情報表示

(3) NS エージェントの再構築を通知する．オリジナルエージェントの位置情報を取得したアクセスエージェントは，新たな NS エージェントとして再構築される．そのため，リンクに対して NS エージェントの変更と，NS エージェントに対する問い合わせの禁止解除を通知する．

本手法により，NS エージェントの消失による位置情報取得の失敗を防ぐことができる．

5. モバイルエージェントビューワ *MiSight*

本手法の応用として，モバイルエージェントに基づくシステム開発支援のためのビューワ *MiSight* の実装について述べる．*MiSight* は，エージェントの動作状況を表示すると共に，アクセスエージェントを介した，遠隔地のエージェントへのスムーズなアクセス機能を提供する．*MiSight* を使うことにより，開発者は他のプラットフォームに移動したエージェントを，ローカルな計算機上のプラットフォーム上で操作できる．具体的には，*MiSight* を利用することで以下の利点が得られる．

- オリジナルエージェントの移動先プラットフォーム内情報の表示
  - エージェントの移動先の情報を開発者が認識しながら，位置透過的な操作が可能
- アクセスエージェントの削除および再生成
  - 開発者はアクセスエージェントの管理が直接可能
- アクセスエージェントの位置管理情報の表示
  - エージェントの移動ルートのログの確認が可能

図 4 に *MiSight* の表示例を示す．*MiSight* では，プラットフォーム内のエージェントをアイコンで表示する．通常のエージェントとアクセスエージェントは異なるアイコンで表示され，指定したアイコンで示されるエージェントのデータベース及び行動ログを閲覧することができる．

また，*MiSight* は Web ブラウザをインターフェースとした情報提示，及びエージェント操作機能を持つ．Web ブラウザを用いた情報表示は，*MiLog* がインストールされていない環境からでも，*MiLog* の情報を閲覧するためのサービスである．Web ブラウザを用いることにより，複数計算機での同時閲覧や遠隔地からの閲覧／操作を行うことが可能である．Web ブラウザによる情報表示では，図 4 で示されるサービスに比べ，表示情報の簡易化以外は同等のサービスを提供する．

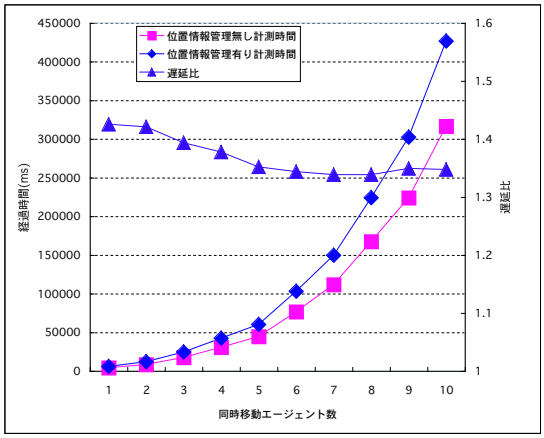

図 5: 遅延比の変化

6. 考察

6.1 評価

本提案手法では，NS エージェントにオリジナルエージェントの位置情報を管理させる．そのため，アクセスエージェントと NS エージェントとの通信時に，システムの動作に遅延が発生する．本提案手法を用いる事により遅延は発生するが，用いない場合と比較した場合の遅延比が，エージェント数が増えても一定であることを示す．遅延比が一定ならば，本手法を用いないアプリケーションを，そのまま本手法を用いて構築可能である．実験として，エージェントに対してアクセスエージェントによる位置情報管理が有る場合と無い場合に分け，各 20 回プラットフォーム間を移動させる．本実験をエージェントの同時移動数を変更して，各条件毎に 100 回の計測を行い，平均結果の比較を行った．本実験では，OS が Windows2000(CPU Athlon 1700+)DOS/V 機 2 台，および eMac(CPU PowerPC G4/1G) 2 台の計 4 台の計算機上で動作させる．全ての計算機は 100Mbps イーサネットでネットワーク接続されており，互いに通信可能である．エージェントは互いに非同期に決まった経路で 4 台の計算機上で動作している *MiLog* プラットフォームを巡回し，特定回数の移動後，経過時間を記録する．

以上の実験の結果を図 5 に示す．図 5 は実験結果をグラフ化したものである．グラフより，遅延はほぼ一定の割合で発生していることがわかる．そのため，本手法を用いた場合の遅延比は一定に保たれるといえる．絶対時間を見た場合，今回の実験環境では，エージェントが単体で移動する場合の遅延時間は 1 回の移動あたり 100ms 未満であり，実際には無視できる程度の遅延である．しかし，同時移動数の増大毎に，絶対時間が増えるため，同時移動するエージェント数が非常に多い場合，遅延が大きくなる．実際には，アプリケーションの構築において多数のエージェントの移動を同時に行うことを避ける等の工夫によって回避可能である．

6.2 特長

本稿で述べた位置透過手法は，以下の特長がある．

- NS エージェントによる位置情報管理により，位置情報取得の負荷集中を抑制できる．
- NS エージェントの再構築機構を備えるため，NS エージェントの消失による障害に耐性がある．
- *MiLog* に限らず，他のモバイルエージェント環境にもアクセスエージェントの実装が可能である．

6.3 関連研究

本稿における提案手法では，モバイルエージェントにおける位置透過的なエージェント操作を実現している．文献 [Takahashi 01] では，エージェントの位置透過の実現の為に，モバイルエージェントフレームワークを構築する際，プロトコルとして実装を行うことによって位置透過性を実現している．本稿では，すでに構築済みのモバイルエージェントフレームワークに対して位置透過性を実現する手法であり，応用として，他のフレームワークに対しても実現可能である．文献 [Numasawa 03] では，位置透過性実現手法として，文献 [Milojicic 98] より forwarding のモデルに着目し，forwarding のもつ問題を多重パスメッセージ転送ネットワークを用いることで解決し，厳密な数理モデルを解析している．本稿の手法は主として registering のモデルを用いているが，NS エージェントを動的に変更する際，forwarding のモデルを一部用いている．

7. おわりに

本稿では，モバイルエージェントの位置透過操作手法を提案した．本提案手法により，モバイルエージェントシステムに基づくシステムの開発および運用が効率化される．エージェントの標準化促進団体 FIPA[FIPA] では，Agent Communication Language（以降，FIPA ACL）や，Agent Platform などの仕様が定められている．FIPA ACL では，メッセージの送信側と受信側エージェントの物理的なアドレスを指定する仕様となっている．すなわち，FIPA に則った Agent Platform では，エージェントの位置を意識して開発を行わなければならない．本稿の手法を用いることで，エージェントの位置情報管理の負担が大幅に削減される．今後の課題として位置情報管理手法の効率化が挙げられる．

参考文献

[Fukuta 00] N. Fukuta, T. Ito, and T. Shintani, ” *MiLog*: A Mobile Agent Framework for Implementing Intelligent Information Agents with Logic Programming, ” First Pacific Rim International Workshop on Intelligent Information Agents, PRIIA2000, pp.113-123, 2000.

[Fuggetta 98] A. Fuggetta, G. P. Picco, and G. Vigna, ”Understainging code mobility,” IEEE, Transactions on Software Engineering, Vol. 24, pp.342-361, 1998.

[Hattori 03] 服部宏充, 大囿忠親, 新谷虎松, ”論理型言語 MiLog に基づくインターネットオークション入札支援システム BiddingBot の実装技法,” 情報処理学会論文誌, Vol. 44, No. 7, pp. 1752-1755, 情報処理学会, Jul. 2003.

[Takahashi 01] 高橋健一, 鍾国強, 雨宮聡史, 峯恒憲, 雨宮真人, ”エージェント基盤のための通信プロトコル：Agent Platform Protocol”, MACC2001, 2001.

[Kubota 98] 久保田稔, ”分散オブジェクト指向システムにおけるメッセージのトレース,” 情報処理学会論文誌, Vol39, No1, pp. 91-101, 1998.1.

[Milojicic 98] D. S. Milojicic, W. LaForge, D. Chauhan, ”Mobile Objects and Agents (MOA)”, 4th USENIX Conference on Object-Oriented Technologies and Systems, April 1998.

[Numasawa 03] 沼澤政信, 栗原正仁, ”多重パスメッセージ転送ネットワークの数理モデルと論理”, 情報処理学会 第 46 回数理モデル化と問題解決研究会, 広島, Sep. 2003.

[FIPA] FIPA : http://www.fipa.org