契約番号 98 第 021 号

高度情報化支援ソフトウェアシーズ育成事業

高階モーバイルエージェントシステムの設計と実装

論文

平成11年2月

（株）インパクト

和文サマリー

高階モーバイルエージェントを実現するフレームワークとして、エージェントの階層的構成とエージェント間移動の二つの概念を提案する。これはモーバイルエージェント内に他のモーバイルエージェントを内包できるものであり、さらにモーバイルエージェントはコンピュータだけでなく、エージェント間を移動できるようになる。論文では、これらの概念に基づいた拡張可能性をもつモーバイルエージェントの実現システムを例示しながら、エージェントが行うコンピュータ間移動、エージェント間通信、永続化などの多様な操作を、エージェント階層構造の変更として統一的に記述・実現できることを示し、また、実現システムに関しては、その構成の最小化や、多様な計算環境への動的適応性や拡張性に有用となることを示す。また、そのシステムの実装についても概説する。

英文サマリー

This paper presents a new framework for higher order mobile agents. The framework provides the notion of agent hierarchy, in which each mobile agent can be a container of other mobile agents, and introduces agent migration among mobile agents as a basic mechanism of the framework. We construct an extensible and portable mobile agent system based on the framework. The system consists of a core system and subcomponents which are implemented through mobile agents. It is characterized by being able to dynamically change and evolve its faculties, including agent migration over networks and agent persistence, by moving agents which offer the faculties. We describe an implementation of the mobile agent system.

研究内容と発注仕様書の対応について

発注仕様書の主な機能と研究内容の対応関係

- 高階モーバイルエージェントのモデルに関しては、2 節で議論され、その実装については 3.1 節で述べる。
- モーバイルエージェントの置換機構は 3.1 と 3.2 節に示される。
- モーバイルエージェントのための協調記述言語は 3.3 節で議論される。

なお、本論文をベースとなった「モーバイルエージェントの階層的な構成と移動」は日本ソフトウェア科学会高橋奨励賞を受賞した。

高階モーバイルエージェントシステムの設計と実装

佐藤 一郎*

〒112-8610 東京都 文京区 大塚 2-1-1
お茶の水女子大学 理学部 情報科学科

1 はじめに

モーバイルエージェント [4] は、コンピュータ間を移動しながら計算処理を継続していく自律的なプログラムであり、その実現システムはすでに数多く提案されている。しかし、既存のシステムでは、エージェントの移動という基本計算メカニズム以外に、永続化や、複製、エージェント間通信などの多様な機能を提供しており、これらがエージェントプログラムの複雑性の原因の一つになっている。また、モーバイルエージェントの動作環境は通常の分散計算より多様化すると予想され、例えば、PC や WS だけでなく情報家電や PDA も含まれる。さらにエージェントのコンピュータ間移送に利用するデータ通信形式も有線だけでなく無線通信となることがある。しかし、既存のシステムでは、特定の OS や通信プロトコルを仮定した構成をとることが多く、それらが利用できない計算環境には適応できない。また、エージェントの計算内容により、システムに提供された以外の多様なサービスや移送方式が要求されることもある。

本論文は、上記の問題を考慮したモーバイルエージェントの基礎モデルとそれに基づくシステム構成を提案するものであり[1]、次節においてそのモーバイルエージェントモデルを提案し、3 節ではそのモデルの実現システムを、4 節では実装概要を述べる。5 節では関連研究を、6 節ではまとめと課題を示す。

2 モーバイルエージェントモデル

ここで提案するモデルは以下の機能をモーバイルエージェントに拡張したものとなる。

*Email: ichiro@is.ocha.ac.jp

[1]同モデルに基づく実装に関しては論文 [7] に譲り、本論文ではシステム構成における基本的な考え方を概説する。また、実装詳細及びプロトタイプシステムは http://islab.is.ocha.ac.jp/agent より参照可能。

- **エージェントの階層的構成:** モーバイルエージェントはその内部に 0 個以上の別のモーバイルエージェントを入れ子状に内包できるとする。また、内包されたエージェントもそれぞれ能動的に動作する。
- **エージェント間移動:** モーバイルエージェントはコンピュータだけでなく、他のモーバイルエージェントに移動できるとし、そのとき、それに内包されたモーバイルエージェントもその階層構造を保ったまま移動するとする。

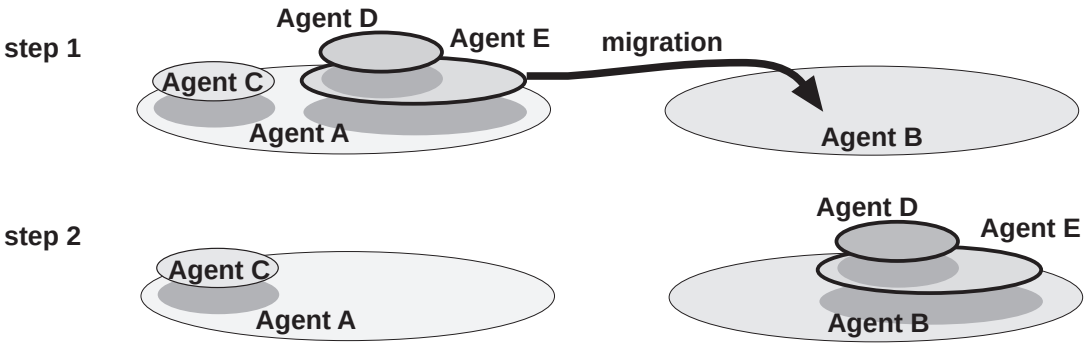

図 1: エージェント階層とエージェント間移動

ここでエージェント階層間の関係を述べる。エージェント階層において、各エージェントは一つ外側に位置するエージェントが提供する各種リソース及びサービスを利用できる。一方、外側エージェントはそれが内包するエージェント階層上の任意のエージェントに対して生成、停止、直列化などの実行制御を行うことができ、また、内包されたエージェントに用意されたコールバックメソッドを適宜呼び出すことができる。なお、内包エージェントが利用する各種サービスや呼び出されるコールバックメソッドは、外側エージェントにより規定される。このため、別のエージェントに移動することにより、利用可能なサービス内容や、コールバックメソッドの呼び出しタイミングや名前を動的に変更できるようになる。

3 モーバイルエージェントシステム

上述の二つの概念の有効性を例示するため、これらを積極的に利用したモーバイルエージェントの実現システムを考える。ここでは、実現システムの核となるランタイムシステムと、それを補助するモーバイルエージェント群の二つからなる構成を考える (図 2)。

3.1 ランタイムシステム

これはモーバイルエージェントの生成、実行、停止を制御し、エージェント階層の管理を行う部分であり、各エージェントは並行実行される。なお、ランタイムシステムは一つ

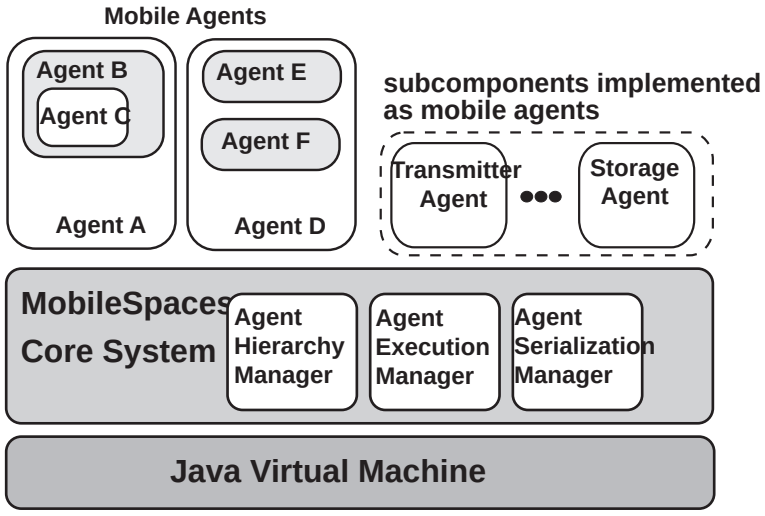

図 2: MobileSpaces のシステム構成

のエージェントとして抽象化され、ランタイムシステム間の通信ネットワークを根とするエージェント階層に位置する。そして、内包するエージェントに対してはエージェントの生成、停止、直列化、階層構造の変更などの基本操作を提供する。モーバイルエージェントに対する他の操作は、モーバイルエージェントとして実現されたシステム補助モジュールにより提供することとし、ランタイムシステムの最小化を実現するとともに、計算環境に依存した操作をランタイムシステムから分離する[2]。

3.2 システム補助用モーバイルエージェント

エージェントに対する操作の一部は、各操作に対応したシステム補助用モーバイルエージェントを通じて実現される。これらはランタイムシステム上に用意されたエージェントであり、到着したエージェントに対して直ちに所定の操作を行う。つまり、操作の対象となるエージェントを、その操作を実現するシステム補助用エージェントに移動させると、その補助用エージェントが到着たエージェントに対してその操作を自動的に行うものである。以下に補助用エージェントの例をあげる。

コンピュータ間移送エージェント : これはエージェントのコンピュータ間移送を実現するモーバイルエージェントであり、送信側 (移送元) と受信側 (移送先) の組として用意される (図 3)。送信側エージェントはそれに到着したエージェント (図 3 ではエージェント A) を直ちに直列化し、エージェントの実行状態とプログラムコードを受信側エージェントに送信する。受信側エージェントではその受信したデータを再びエージェントに戻す。

[2]現在の実装では、便宜上の理由から、ランタイムシステムは TCP 通信を利用したコンピュータ間のエージェント移送機能を提供する。

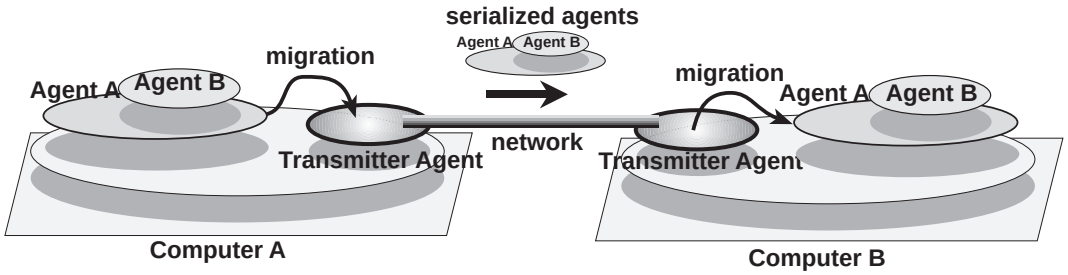

図 3: コンピュータ間移動エージェント

なお、両エージェント間のエージェント移送において、TCP や UDP などのデータ転送プロトコルの選択や、移送における通信手順は両エージェントにより定義される。そして、別のコンピュータ間移送エージェントに移動することにより、移送方式を動的に変更できるようになる。

回送エージェント : エージェントが他のエージェントへの移動を試みたとき、その移動先のエージェントがすでに他に移動していることがある。このとき、移動するエージェントは移動元に回送エージェントを残すことにより対処する。回送エージェントは移動したエージェントの識別子を受け継ぎ、それに到着したエージェントを、その移動したエージェントに直ちに移送するエージェントである。これはエージェント階層のショートカットとしても機能する。

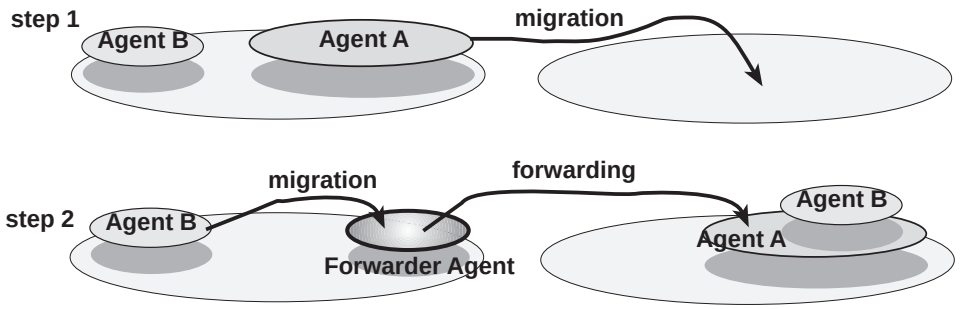

図 4: 転送エージェント

この他、エージェントの永続化や複製などの操作は、上記の移送エージェントと同様にその操作を実現するエージェントに移動することによって実現される。また、エージェント間通信は、通信先のエージェントに直接移動する方法と、内包エージェント間の通信サービスを提供するエージェントに移動させる方法がある。

ここで提案したモーバイルエージェントシステムはそのシステム構成の最小化以外にも次の利点がある。

- エージェントに対する各種操作を、移動というモーバイルエージェントの基本的な

計算メカニズムとして統一的に取り扱うことができるため、エージェントプログラムの複雑性を軽減させることができる。

- 各種機能がモーバイルエージェントとして提供されるため、新たな機能を提供するモーバイルエージェントを移動させることにより、システムの動的拡張を行うことができる。

- モーバイルエージェントの利点の一つは、特定のコンピュータに束縛されずに実行できることにある。このため、エージェントに対するすべての操作をコンピュータ（ランタイムシステム）それぞれに用意する必要なく、特定のコンピュータにおいてのみ実現し、そこに移動して操作を行えばよい。

- コンピュータ（ランタイムシステム）もモーバイルエージェントとして扱えることから、携帯端末などの移動性をもつコンピュータについてもモーバイルエージェントとして統一的に扱える。

3.3 移動先エージェントのスクリプティング記述

MobileSpacesシステムの計算機構的な特徴は、コンピュータ間移動や永続化、エージェント間通信などの各種機能をエージェント階層内のエージェント間移動として実現することにある。これにより、多様な計算をエージェント間移動として統一的に記述できることになる。一方、エージェントの移動先の指定はURLs (Uniform Resource Locations)により実現される。そこで、MobileSpacesシステムではこのURLにスクリプティング言語機能を導入する。

エージェントの特定

各エージェントは、グローバルな識別子と、階層構造に依存したローカルな識別子の二つをもち、後者はURLにより記述される。例えば、some.where.com/agent1/agent2では、some.where.comはエージェントの属するホストの名前であり、/agent1/agent2とは親エージェントagent1の子エージェントとなるagent2を示している。なお、各エージェントはエージェント階層内の他のエージェントに、自分自身または子孫エージェントを移動させることができる。

```
go(new AgnetURL("/agent1/agent2"));}
```

上記において/agent1/agent2"));は、これを実行したエージェントが自分自身をエージェント階層内の/agent1/agent2となるエージェントに移動させることを意味する。

```
go(new AgentURL("transmitter:some.where.com/agent1/agent2"));
```

また、コンピュータ間移動や永続化などの計算環境に依存したサービスは、各サービスごとにエージェントが用意されている。従って、これらのサービスを提供するエージェントを選択することより、計算環境の変化に柔軟に対応することができる。一方、計算環境の変化は、MobileSpacesのランタイムシステムではなく、オペレーティングシステムなどの下位レベルのソフトウェアにより監視・通知される。そこで、オペレーティングシステムとMobileSpacesのインターフェースを導入する。ただし、MobileSpacesは多様なオペレーティングシステムやハードウェアで稼働することを目的としているため、オペレーティングシステムへの変更は必要ない方法とする。

エージェントの指定するURLは形式`$(variable)`で書かれた変数を含めることができる。この変数はオペレーティングシステムやシェルによって提供される環境変数に対応づけられる。これは動的環境変数 [?] 及び動的 URLs [?] と同様なものとなる。

```
$(TRANSMITTER)://some.where.com/agent1/agent2
```

上記では`TRANSMITTER`は環境変数に対応づけられた変数であり、その中身はコンピュータ間移動サービスを提供するエージェントの名前となる。そして、計算環境の変化に応じて、コンピュータ間移動に最適なエージェントの名前がその内容となる。

4 実装・評価

ランタイムシステムはJava言語(JDK1.1以上)のアプリケーションプログラムとして実装され、エージェントプログラムもJava言語のオブジェクトとなる。同言語処理系が稼働する環境であればOSやハードウェアに依存せず、エージェントの実行・移動が可能となり、計算環境に依存した部分は補助的エージェントにより提供できる。

ランタイムシステム

エージェント階層を維持管理すると同時に、各エージェントに一つ以上の実行スレッドを割り当てて能動的な実行を実現する。また、コンピュータ間移動や永続化などで利用するエージェント／オブジェクトの直列化はJava言語の直列化機構を利用して実現する[3]。

モーバイルエージェント

図5のように構成され、主な要素は次の通り。

- **エージェントプログラム**
  各エージェントの動作を記述するJava言語のプログラムであり、外部エージェン

[3]エージェントの永続化やコンピュータ間移送の対象なるのはヒープ領域内の状態となる。この他、エージェントのプログラムコード、エージェントの階層構造、識別子なども移動・永続化の対象となる。

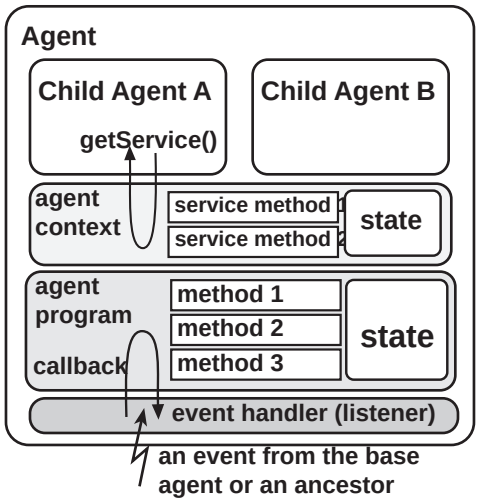

図 5: 階層モーバイルエージェント

トからの呼び出されるコールバックメソッドと、内包エージェントが呼び出すサービス用のメソッドから構成される。各メソッドプログラムには、外部エージェントへのサービス要求や内包エージェントへのコールバックメソッド呼び出しを含むことがある。

- **エージェントコンテキスト**
  内包されるエージェントに対して各種サービス、例えば、生成、停止、エージェント間移動などを提供するための API 群からなり、エージェント識別子や階層構造を含むリソース情報も提供する。なお、外側のエージェントが提供するサービス用メソッドも、エージェントコンテキストを介して呼び出す。
- **コールバックモデル**
  内包側エージェントに対して呼び出すコールバックメソッドの種類とタイミングを規定する部分である。内包エージェントの状態遷移モデルとコールバックメソッドを対応づける Java 言語プログラムにより定義される。
- **内包エージェント**
  あるエージェントに内包されている 0 個以上のエージェントであり、能動性をもち、さらにこれの内部に入れ子状にエージェントを含むことがある。

この他、エージェントは、それが属している階層構造に従って与えられる相対的な識別子と、実行に必要なプログラムコードを保持する。なお、基礎的な性能評価としてエージェントの移動速度を表 1 に示す。ただし、測定環境は、Sun 社の JDK(v1.1.6)、CPU は IBM-PC/AT 互換機 (PentiumII:400MHz)、エージェントの移送には Ethernet(100BASE-TX) ネットワーク上で TCP 通信を利用したものであり、同一ランタイム内の移動ではエージェントの一時停止や直列化は行わないとする。

表 1: エージェントの移動時間 (参考値、単位は秒)

相違なコンピュータ間の移動時間	0.3s
同一ランタイム内の移動時間	0.05s

5 関連研究

前述のように数多くのモーバイルエージェントシステムが提案されているが (例えば、Telescript[8]、Aglets[3]、Voyager [5]、AgentSpace [6] など)、モーバイルエージェントの階層化を導入した実装システムは皆無であり、Telescript などではプレースと呼ばれるエージェントによる複数エージェントの内包や、エージェントのグループ化 [1] の試みはあるが、プレースやグループ自体に移動性はない。一方、オブジェクトを基礎とするものには具体的な実装事例がいくつか存在している。例えば、Aperios [9] はオブジェクト移動を提供する OS であり、オブジェクトを特定のメタオブジェクトに移動させることによって動作内容の動的変更を実現することができる。

6 おわりに

この論文では、階層化及びエージェント間移動の二つの概念をもつモーバイルエージェントモデルを提案し、それの実現システムについて議論した。しかし、数多くの課題も残されている。例えば、エージェントがもつ識別子は階層構造に依存しており、エージェントやコンピュータの移動を含む階層構造の動的変更に柔軟に対応できない。また、エージェント自体が計算環境の相違を判別しそれに対処する方法や、セキュリティ機構も必要となる。また、この論文で提案したエージェントモデルは、プログラムの移動性を考慮した並行計算理論 Mobile Ambients[2] に対応したものとなるが、同理論に基づく本モデルの形式化が重要な課題となる。

参考文献

[1] J.Baumann and N.Radounklis, *Agent Groups in Mobile Agent Systems*, Conference on Distributed Applications and Interoperable Systems, 1997.

[2] L.Cardelli, and A.D. Gordon, *Mobile Ambients*, Proceedings of Foundations of Software Science and Computational Structures, LNCS 1378, Springer, 140-155. 1998.

[3] IBM Tokyo Reserach Laboratory, *Aglets Workbench: Programming Mobile Agents in Java*, http://www.trl.ibm.co/aglets, 1997.

[4] D.B.Lange, *Mobile Objects and Mobile Agents: The Future of Distributed Computing?*, Proceedings of ECOOP'98, LNCS 1445, Springer, pp.1–13, 1998.

[5] ObjectSpace Inc, *ObjectSpace Voyager Technical Overview*, ObjectSpace, Inc. 1997.

[6] I.Satoh, *AgentSpace: A Mobile Agent System*, http://islab.is.ocaha.ac.jp/agent/index.html, 1997.

[7] 佐藤一郎, 高階モーバイルエージェントシステム, 情報処理学会プログラミング研究会報告, 3 月, 1998.

[8] J.E.White, *Telescript Technology: Mobile Agents*, General Magic, 1995.

[9] Y.Yokote, *The Apertos Reflective Operating System: The Concept and its Implementation*, Proceedings of OOPSLA'92, pp.414-434, 1992.