マルチメディア通信と分散処理 88－5
（1998．4．23）

Javaによるモバイルエージェントシステムの実装

三好 優[*1] JUNG Euihyun[*2] JIE Jung Hoon[*2]
朴 容震[*2*3] 浦野 義頼[*3*4] 富永 英義[*1*3]

[*1] 早稲田大学理工学部 電子・情報通信学科
[*2] 漢陽大学校 工科大学 電子工学科
[*3] 通信・放送機構早稲田リサーチセンター
[*4] 早稲田大学理工学総合研究センター

モバイルエージェントの特長は相手先のホストに実行可能なプログラムが輸送されて処理を行う”マイグレーション機能”にある．そのため，相手先のホストと常に接続を維持している必要がなく，ネットワークトラヒックの低減が期待される．また周波数帯域などの環境が変化し，しばしば通信コネクションが切断される，モバイルコンピューティング環境下でも実現できる．

本稿では，このようなエージェントの考え方を用いることによってプラットフォームやOSに依存しないシームレスなコンピューティングを実現するモバイルエージェントシステムの実装を行い，またそのアプリケーションとして，ネットワーク上の動的情報の収集・管理をサポートするネットワーク管理エージェントを提案し，システムの実装を行った．

キーワード Java, エージェント, モバイルエージェント, MOBACS, モバイルコンピューティング, ネットワーク管理

An implementation of Mobile Agent System with Java

Yu MIYOSHI[*1] Euihyun JUNG[*2] Jung Hoon JIE[*2]
Yong-Jin PARK[*2*3] Yoshiyori URANO[*3*4] Hideyoshi TOMINAGA[*1*3]

[*1]Dept. of Elec., Info. and Comm. Eng., WASEDA University
[*2]Dept. of Elec. Eng., HANYANG University
[*3]TAO Waseda Research Center [*4]ARISE, WASEDA University

The outstanding characteristic of a mobile agent system is “migration” when executable programs are transmitted from the source platform and executed at the destination platform.

Since we need not constantly maintain the connection during communication, network traffic would be easily reduced. Furthermore, agent-based communication is appropriate for mobile computing environment where available frequency band dynamicaly changes and it is difficult to maintain connections.

We have implemented an OS-independent mobile agent system with Java and proposed the network-management agent collecting network-oriented information.

keywords Java, agent, mobile agent, MOBACS, mobile computing, network management

1. はじめに

近年エージェントと呼ばれる技術が人工知能をはじめとする様々な分野で研究され，急速に発展している．情報通信の分野におけるエージェントは，人間が行っていたネットワークにおける複雑な処理を代行し，そのサポートを行う．また，モバイルコンピューティングなどの環境のダイナミックな変化にも柔軟に対応し，その知能によって分散協調処理を可能とする．

その概念の理想は，環境やプラットフォームを始めとするあらゆる壁を越えて通信を実現する究極のコミュニケーションサポートといえる．エージェント技術が分散情報処理，モバイルコンピューティングに一般的に利用されるようになれば，従来の環境は大きく変貌することとなり，既存の通信サービスを劇的に変え得るものとして期待される．

本研究ではエージェントの考え方を用いることによって，さまざまな制限を持つ従来のモバイルコンピューティング環境においても通常のコンピューティングと同様に機能し，プラットフォームやOSに依存しないシームレスなコンピューティング環境を提供するモバイルエージェントシステムの構築を行っている．また，構築したモバイルエージェントシステムの利点を活かしたアプリケーションとしてネットワーク管理エージェントを提案し，その実装を行った．

2. MOBACS[1]

MOBACS(Mobile Agent Computing System)は，実際のネットワーク上にモバイルエージェントが動きまわるエージェントネットワークを仮想的に創り出している．モバイルエージェントは各ホスト上に構築されたCell(図1)と呼ぶエージェントエンジン上で動作し，Cellを通して互いに通信を行うことができる．

本システムは全てJavaで記述されている．そのためモバイルエージェントはJavaの実行環境を備えたホストであればプラットフォームやOSに左右されずどのような環境であっても同様に動作する．

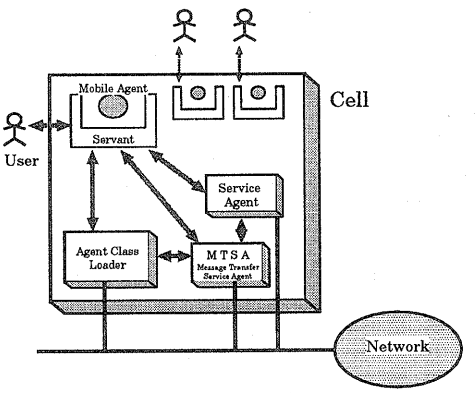

図 1: Cellの構成

2.1 Agent/Servant構造

MOBACSでは，Java appletによってモバイルエージェントを実装している．しかし，Javaのセキュリティモデルはappletによる移動先のホストのローカルな資源へのアクセスを禁止している．そこでAgent/Servant 構造を新しく導入し，問題を解決した．

すなわち，Agent/Servant構造では，モバイルエージェントがホストのローカルな資源にアクセス要求すると，Servantはそのアクセス要求が正当であるかを検証し，正当な場合のみ要求の実行を行う．ServantはJavaアプリケーションとして実装されているので，ローカルホストのリソースへのアクセスが可能である．

2.2 マイグレーション

本システムにおけるモバイルエージェントのマイグレーションはCell間のマイグレーションプロセスによって行われる．この時同時にプロセスの実行コードやデータが移動することが必要である．以下にMOBACSにおけるモバイルエージェントのマイグレーションプロセスを示す(図2)．

モバイルエージェントのマイグレーション時，送出側のServantが目的地のCellにメッセージを出す．それを受けて目的地側のCellがServant を創り出して待機する．その後送出側のServantはエージェントコードやデータをパッケージ化して送出し，目的地側にモバイルエージェントのコードが到着するとモバイルエージェントの移動が完了する．このプロセス

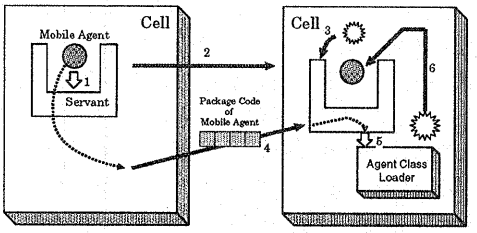

図 2: マイグレーションプロセス

を繰り返すことによりモバイルエージェントが各ホストを次々とマイグレイトすることが可能となる.

3. アプリケーションシステムの構築

本項では実装・改良を行ったMOBACSを基に，システムが持つマイグレーション機能を有効に活用するためのアプリケーションを提案する.

3.1 ネットワーク管理エージェント

ネットワーク上には大量な情報が散らばっている．これらの情報はさまざまなフォーマットで記述され，また日々変化し，増加し続けるものも少なくない．そのように動的に変化する情報を一人一人のユーザが自分の思うように効率的に利用することは困難である．例えばネットワーク管理のためにその時の各ホストのトラヒック状況，CPUの能力などを知りたいだけであってもシステム管理者はそのたびにリモートログインを繰り返し，接続を維持しなければならない.

そこで，モバイルエージェントのアプリケーションの一つとして，ネットワーク上に分散し蓄積され続ける様々な動的ネットワーク情報に対して，一人一人のユーザにカスタマイズした処理を行うことによって効率良い情報収集のサポートを行うネットワーク管理エージェントを提案する.

ネットワーク管理エージェントは，モバイルエージェントを用いてユーザの望む情報をネットワーク上から探し出し，カスタマイズして提示する．その際，ユーザはエージェントの提示した情報がどのように得られているのかを知らずとも，より少ない手間で必要な情報を手に入

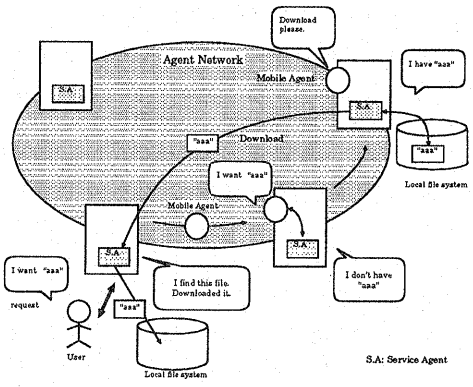

図 3: ネットワーク管理エージェントの概念

れることができる.

また，情報先のホストあるいはユーザが使用しているホストが，どのようなOS，プラットフォーム，環境(モバイルコンピューティングなど)であろうと同様に動作するため，あたかもネットワークが一つのコンピュータであるような情報管理・操作を実現する.

例えばユーザがあるファイルを指定すれば，そのファイルがどのホスト上にあるかを指定しなくても(知らなくても)モバイルエージェントがファイルを探し出してダウンロードまでの一連の処理を代行する．また，それぞれのホストの能力をエージェントシステムに示しておけば，能力に合わせて情報にフィルタをかけてダウンロードを行うといった処理が可能になる．また，CPUの能力，ネットワークのトラヒック，ホストが処理できる情報の種類などをエージェントシステムに渡せば，ユーザが指示した処理をどのホストで行えば適当か判断を行う.

ネットワーク管理エージェントを利用することにより，ユーザは単純な命令のためにリモートログインをする必要がなくなる．リモートホストの持つ必要な情報をエージェントに取りに行かせる命令をしておけば良い．本システムではリモートホストにおけるユーザの処理そのものの代行も可能である.

筆者らはこのネットワーク管理エージェントをモバイルエージェントシステム上で実現するため，必要と考えられる機能の実装を行った.

3.2 システム内における情報のやりとり

従来のMOBACSで実装されているサービスを行う機能を持ったエージェント(サービスエージェントと呼ぶ)はMTSA(Message Transfer Service Agent)のみである．MTSAはユーザからServantへ渡されたメッセージを，マイグレイトするモバイルエージェントのコードにデータとして付加する．また，ネットワークを通ってきたモバイルエージェントのコードから渡されたメッセージを取りだし，表示する．これによりMTSAはエージェントネットワーク上でモバイルエージェントによるメッセージのやり取りを行う機能を実現している．

ネットワーク管理エージェントには，ローカルホストに格納されたファイルや，ローカルホストが提供する動的ネットワーク情報をモバイルエージェントに集めさせる機能が必要であり，エージェントネットワーク上で情報のやり取りを行う機能を何らかの方法で実装する必要がある．そこでシステム内で情報のやり取りを行うため，本システムでは二つの情報授受モードが実装されている．

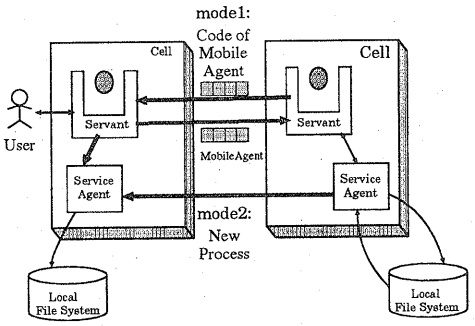

図 4: システム内の情報授受モード

モード1．モバイルエージェントに各ホストでのアプリケーション実行結果を付加する情報のやり取り

モード2．各アプリケーションにユーザへの実行結果の返信を任せる情報のやり取り

ネットワークトラヒックを考慮すると，ネットワーク上を移動するモバイルエージェントには，あまり多くの情報を付加するのは好ましくない．しかしマイグレイト先のホストで得られる情報を次のホストに持ち込むにはモバイルエージェントに情報を持たせる必要がある．ネットワーク情報管理エージェントは，動的情報の収集を行う場合にはモバイルエージェントへの情報の付加，またファイルのダウンロードを行う場合には新しいプロセスを作るというように二つのモードの組み合わせによって実現される．

モード1は，各ホストでのアプリケーションが返す実行結果の情報量が少ないサービスの場合，実行結果を返さないサービスの場合に都合が良い．また，各ホストでの実行結果をユーザがまとめて得たい場合，あるホストの実行結果を用いて次にマイグレイトしたホストで実行するといった分散コンピューティングを行う場合に必要である．

一方モード2は，ネットワーク上に散在するファイルなどのリソースをモバイルエージェントを介さずにそのまま得る場合や，一つ一つの処理に極度の時間が必要となり一つの処理を行っている間に別の処理をさせたい場合に有効となる．

ネットワーク情報管理エージェントは両モードの特徴を考慮し，両モードを組み合わせることで実現されるべきである．両モードを複合的に用いることによって，より効率的に分散した動的ネットワーク情報の収集を行うことが可能となる．

3.3 サービスエージェント

本項ではネットワーク情報管理エージェントを実現するためにその機能の一部としてインプリメントした各サービスエージェントについて述べる．

ネットワーク情報管理エージェントのために実装した各サービスエージェントは，基本的にはその動作時にMTSAを通じてサービスを行っている．これはMTSAが提供するサービスであるメッセージを，ユーザからのコマンドとして扱うことによって各サービスエージェントが動作するようにシステムを設計したためで，ユーザはメッセージ(コマンド)と，リモートホストのIPアドレスを入力することに

よって，モバイルエージェントに指示を出すことができる．

メッセージをコマンドとして使用することでJavaのマルチプラットフォーム性を高めることができる．命令をあらかじめインプリメントしておく場合，その命令に対する機能が存在しないプラットフォームにおける動作が保証されない．そこでユーザのメッセージをコマンドとして使用することによって，そのプラットフォームに存在しない命令の場合にはただのメッセージとして送られることになり，ユーザがモバイルエージェントの移動先のプラットフォームを意識しないための設計となっている．

ここで，実装した各サービスエージェントについて述べる．

FTAgent(File Transfer Agent)

本システムでは，ユーザが指定したファイルをダウンロード(モバイルエージェントから見た場合はアップロード)するサービスエージェントをFTAgentと呼んでいる．FTAgentを利用した場合の情報授受モードにはモード2が選ばれる．モード2はFTAgentの一部としてインプリメントされているので，モード2を利用する際には必ずFTAgentを用いることになる．またユーザが指定したファイルを探す機能も備えている．

コマンド実行エージェント

ユーザからのメッセージをコマンドとして扱うことによって，リモートログインすることなくそのホストのプラットフォームで提供されている機能のほぼ全ての処理を行うことができるサービスエージェントである．しかし，全ての処理が実行されてしまうことの危険性を考慮して，しかもシステムの安全性を求めるために現在は指定したいくつかの処理だけを行うように設定してある．主にネットワーク関連のステータスを集めてユーザに提示する機能として実装されている．

GUI表示

ネットワーク情報管理エージェントがユーザに返す情報は，GUIコンポーネントによって図5のように表示される．

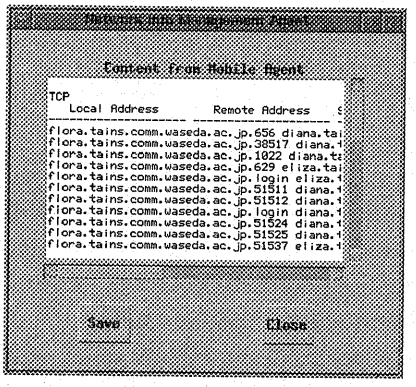

図 5: 処理結果

4. MOBACSに実装されているその他の機能

4.1 Servantの個人化

MOBACSではエージェントIDという概念を導入し，CellがエージェントIDごとにServantを作ることによってモバイルエージェントのカスタマイズが実現される．

ネットワーク管理エージェントにおいても，エージェントIDを延長してモバイルエージェント(Servant)のさらなるカスタマイズを行うことを検討した．

ユーザは本システムを利用する場合，必ず特有なエージェントIDを入力する．そのエージェントIDごとに，ユーザが指定するモバイルエージェントの行き先のホストのIPアドレスや行動を記録するユーザプロファイルを作成し，次回モバイルエージェントが利用される場合そのデータを参照することによってカスタマイズを図っている．

```
133.9.73.3   :   search&Servant.java
133.9.71.9   :   netstat
133.9.73.7   :   netstat
133.9.73.3   :   finger
133.9.73.8   :   back
```

図 6: 作成されたプロファイル例

また，ユーザがネットワーク管理エージェントを利用して得たデータをローカルファイルシステムに記録する場合，データはユーザごとに

特有のファイルに格納され，これもユーザに対するカスタマイズのためのプロファイルとして用いられる．

4.2 モバイルエージェントの自律性

従来のMOBACSにはエージェントの大きな特徴の一つである自律性(Autonomy)が十分には備わっておらず，ユーザの明示的指示がなければエージェントは動作しない．そこでシステムに自律性を持たせ，モバイルエージェントの動作を自動化することを検討した．

現在本システムのモバイルエージェントにはユーザがエージェントに繰り返し同じ命令をエージェントに与えなくともユーザプロファイルを参照し，その情報に基づいて同じ動作を自動的に行う機能が追加されている．

この機能により，モバイルエージェントはユーザからの命令なしで次々とネットワーク上の各ホストをマイグレイトし，処理を行うことが可能である．

4.3 モバイルエージェント間の通信機能

本システムは，モバイルエージェント及びServantと各端末の静的サービスエージェントが情報のやり取りを行うことによってモバイルエージェントとして動作する．この拡張としてモバイルエージェント同士の通信機能を追加する．これは将来本システムがインテリジェンスの伴う協調・ネゴシエーションを行うために必要な機能である．

本システムは一般的に論じられるモバイルエージェントとは次の点で異なった構造をしている．

- エージェントのプラットフォームであるCellがあらかじめ各ホストで動作している
- モバイルエージェントの実行機能をServantが代行している

このためモバイルエージェント同士の通信方式においても他のシステムとは異なる方式が必要となることが考えられる．

モバイルエージェント同士の通信方式にはエージェント間で直接通信を行う方法と中継方式の二つが挙げられる[2]．本研究では，中継方式のなかでも”黒板型”の通信方式を実装する．既に提案されているいくつかのエージェント間通信のなかから黒板型の方式を選択した理由としては，本システムの構造が挙げられる．本システムでは，モバイルエージェントが動作するためにあらかじめCellが各ホストにインプリメントされ，実行されている．つまりモバイルエージェントが移動する各ホストには必ずCellが動作している．このCellを拡張し黒板の役割を持たせることで円滑にモバイルエージェント間通信を実現することができる(図7)．

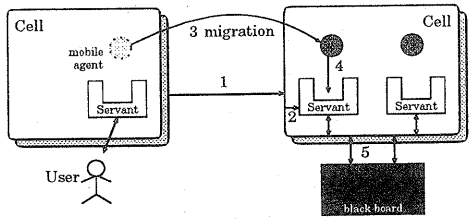

図 7: MOBACS上でのエージェント間通信

5. まとめ

本研究ではモバイルエージェントシステムを構築するためにMOBACSの実装・改良を行った．またモバイルエージェントの利点を活用するアプリケーションとしてネットワーク管理エージェントを提案し，必要な機能の実装を行った．

今後はモバイルエージェントにより高度のインテリジェンスを付加し，複数のエージェントのネゴシエーションの結果ネットワークの管理・運用が為される，分散協調形ネットワーク管理への適用を検討する．

参考文献

[1] Euihyun Jung, Yong-Jin Park, Chulhye Park：Mobile Agent Network for Supporting Personal Mobility, ICOIN-12, Tokyo(1998-01)

[2] 國頭 吾郎, 奥村 恭弘, 相澤 清晴, 羽鳥 光俊：協調エージェント間通信のためのTracking Agentに関する検討：信学技報 IN97-20(1997-04)