モバイルコンピューティングとワイヤレス通信 20−17
(2002. 3. 8)

モバイル環境向けエージェント移動制御

川上 憲治† 広重 一仁† 佐々木 宏† 岡宅 泰邦† 本位田 真一‡

†日本テレコム株式会社 情報通信研究所
〒104-0032 東京都中央区八丁堀 2-9-1

‡国立情報学研究所 / 東京大学
〒101-8430 東京都千代田区一ツ橋 2-1-2

E-mail: †{kawakami, hirosige, chacha, okataku}@japan-telecom.co.jp, ‡honiden@nii.ac.jp

あらまし 近年，携帯電話を始めとするモバイル端末が普及し，モバイル環境の裾野は急速に広がった．これに伴い，モバイル環境においても固定の環境と同じようにアプリケーションを利用したいという要求が増えつつある．しかし，モバイル環境には，モバイル端末のリソース(メモリ，CPU，帯域幅など)の制限や移動に伴うアプリケーション実行環境の変化といった課題があり，先の要求を実現するためにはこれらの課題を解決するための新たな機構が必要になる．筆者らは，その機構を実現するために，モバイルエージェントを基盤とするミドルウェアを提案した．本稿では，提案したミドルウェアの実現に必要なモバイルエージェント移動制御方式について検討している．

キーワード モバイル環境，モバイルエージェント，移動制御

Agent Migration Control for Mobile Environment

Kenji KAWAKAMI†, Kazuhito HIROSHIGE†, Hiroshi SASAKI†, Yasukuni OKATAKU†, and Shinichi HONIDEN‡

†Information and Communication Laboratories, Japan Telecom co., ltd
Hatchobori 2-9-1, Chuo-ku, Tokyo, 104-0032 Japan

‡National Institute of Infomatics and The University of Tokyo
Hitotsubashi 2-1-2, Chiyoda-ku, Tokyo, 101-8430 Japan

E-mail: †{kawakami, hirosige, chacha, okataku}@japan-telecom.co.jp, ‡honiden@nii.ac.jp

Abstract In recent years, users of mobile terminals including cellular phones are growing rapidly, and the use range of mobile applications has greatly spread. In proportion to the spreading, the demand that the users want to use the applications in mobile environments as in the fixed ones is increasing. In mobile environments, however, there are problems to be solved for adapting to limited resources (memory, CPU, bandwidth, and so on) on mobile terminals and changes of the environments in which applications are executed. In order to realize these demands as mentioned, a new mechanism is necessary for solving these problems. In order to realize the mechanism, we proposed a mobile agent-based middleware. In this paper, we consider a migration control mechanism of mobile agents for realization of the proposed middleware.

Key words Mobile Environment, Mobile Agent, Migration Control

1. はじめに

近年，携帯電話を始めとする通信機能を備えたモバイル端末が普及し，モバイル環境の裾野は急速に広がった．これにより，利用場所を選ばずに電子メールを交換したり，インターネット上の Web サーバから必要な情報を検索したりすることが非常に身近になった．さらに今後は，モバイル環境においても固定の環境と同じようにアプリケーションを利用できることが望まれる．

ところで，携帯電話や PDA などのモバイル端末をアプリケーション実行環境という観点から見た場合，その特徴としてアプリケーションから利用できる資源(メモリ，CPU，帯域幅など)が限られることや，端末の移動にともないアプリケーションの実行環境が変化することが挙げられる．

モバイル端末において，固定の環境と同じようにアプリケーションを実行できる環境を実現するには，アプリケーションが利用可能な資源を柔軟に活用でき，実行環境の変化にも適応できる新たな機構が必要となる．

筆者らはその機構として，アプリケーションを実行する環境の変化に合わせて，アプリケーションの処理をダイナミックに変更する機構について検討してきた．そして，それを実現する手段の 1 つとして，モバイルエージェントを基盤とするアプローチを提案した[1]．ここでは，モバイルエージェントは他の端末を利用して処理能力の不足を補う等の役割を果たす．また，提案方法は汎用性や開発効率を考慮し，機構を個々のアプリケーションではなく，ミドルウェアのようなアプリケーションの下位レイヤに実装することを特徴としている．

本稿では，モバイルエージェントを基盤とするミドルウェアの実現に必要な，具体的なモバイルエージェント移動制御方式について検討している．

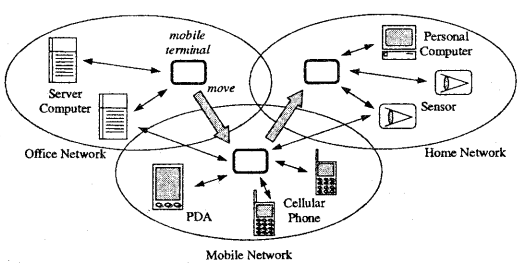


図 1　想定するモバイル環境

2. モバイル環境におけるアプリケーションの課題

2.1 想定する環境

まず，本稿が解決しようとする課題について述べるため，モバイル端末を携帯したユーザが複数のネットワーク間を移動する図 1のようなモバイル環境を想定する．ユーザは，帯域幅や通信コストが異なる複数のネットワークを利用する．ユーザが携帯するモバイル端末には複数のアプリケーションがインストールされており，ハイエンドなサーバからローエンドなセンサにいたる様々な処理能力をもったコンピュータ上のアプリケーションと通信を行う．ユーザの移動にともないモバイル端末が他のネットワークに移動すると，アプリケーションが利用できる帯域幅や通信可能なコンピュータは変化する．このような環境において利用されるアプリケーションの課題について，モバイル端末の制約とアプリケーション利用環境の変化の 2 点から考察する．

2.2 モバイル端末の制約

モバイル端末では，バッテリの持続時間や携帯性のため，CPU やメモリといったアプリケーションから利用できる資源が制限される．今後，モバイル端末に搭載される CPU の性能向上やデバイスの高機能化が進むことが予想されるが，固定の環境で利用されるコンピュータと比べて処理能力の本質的な差異は存在し続けるであろう．しかし，このような端末の処理能力差から利用できるデータやアプリケーションが限定されてしまうと，モバイル端末の利便性は大きく損なわれることになる．

このため，モバイル端末上で実行されるアプリケーションは，単にプログラムサイズを小型化するだけではなく，他のコンピュータの豊富な計算資源やネットワーク資源などを柔軟に利用することにより，限られた処理能力を補うことが考えられ，それを実現する機構が求められている．

例えば，CPU 性能が非常に限られたモバイル端末が高負荷な演算処理を実行しなければならない場合(シナリオ 1)には，他の高性能な端末で処理を実行させることで演算時間を短縮することができる．また，通信帯域が限られたモバイル端末がネットワークを介して他の端末上の大量なデータを処理するような場合(シナリオ 2)には，データが存在する端末上でデータを直接処理することで通信処理を大幅に削減することができる．4.3節では，これら 2 つのシナリオを例に提案方式の動作について説明する．

2.3 アプリケーション利用環境の変化

モバイル端末が異なるネットワーク間を移動する場合，帯域幅や通信コストといったネットワーク環境や通信可能な端末が変わる．また，各端末においても利用可能な計算資源やバッテリなどの資源が刻々と変化する．

このような環境の変化がアプリケーションの実行中に発生するような場合には，その変化に合わせて処理をダイナミックに変更することにより，処理を効率的に実行できる．例えば，前述のシナリオのように他の端末の資源を利用して処理を実行する場合には，このような環境の変化を検出し，かつ最も効率的に処理を実行できる端末を動的に選択することが求められる．

3. ミドルウェアによるアプローチ

3.1 モバイルエージェントを基盤とするミドルウェア

前章で述べたモバイル環境におけるアプリケーションの課題を解決するためには，次に示す機構が必要となる．

- 他の端末の資源を利用してアプリケーションの処理を効率的に実行する機構
- 環境の変化に合わせてアプリケーションの処理をダイナミックに変更する機構

汎用性や開発効率の観点から，これらの機構はミドルウェアとして実装されることが望ましい．しかしながら，従来のミドルウェアはモバイル環境の特徴を考慮していないため，新たなアプローチが必要となる[2].

我々は上述の機構を実現するために，モバイルエージェントを基盤とするミドルウェアによるアプローチを採用する．提案方式で用いるミドルウェアは，複数のモバイルエージェントで構成される．それらのモバイルエージェントが状況の変化に応じて端末間を移動することにより，上述の機構を実現する．なお，モバイルエージェントの移動先は，各端末の状態を総合的に判断するために，各端末上のエージェント間の交渉により決定される．

3.2 アーキテクチャ

図 2にモバイルエージェント(以下，MA と略す場合あり)を基盤とするミドルウェアのシステム構成を示す．提案方式で用いるミドルウェアは，管理 MA 層，サービス MA 層，MA ランタイム層から構成される．

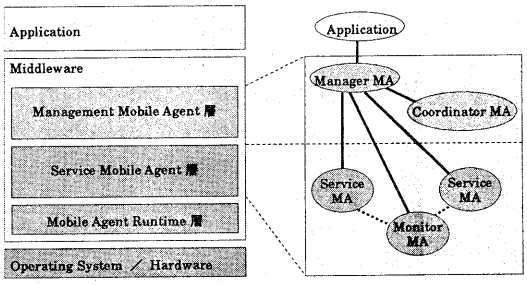


図 2 システム構成

管理 MA 層は，マネージャ MA とコーディネータ MA で構成される．マネージャ MA は，アプリケーションとモバイルエージェントもしくは異なるモバイルエージェント間のブローカとして機能する．コーディネータ MA は，各端末のコーディネータ MA と協調して動作し，サービス MA の移動先端末を決定するための交渉を行う．

サービス MA 層は，複数のサービス MA と複数の監視 MA で構成される．各サービス MA は，通信やデータ入出力など，アプリケーションから利用される個々の機能を提供する．提案方式で用いるミドルウェアはこれらのサービス MA の移動を制御することにより，環境の変化に対するダイナミックな適応を実現する．各監視 MA は，アプリケーションまたは他のモバイルエージェントから利用される実行環境(CPU 能力，帯域幅，メモリ容量，ユーザ位置等)を監視するための個々の機能を提供する．

MA ランタイム層は，モバイルエージェントの実行を制御するための基本的な機能を提供する．

4. モバイルエージェント移動制御方式

提案方式で用いるミドルウェアは，アプリケーションから独立した機構でサービス MA の移動を制御する．本章では，環境の変化に合わせて，サービス MA を適切な端末(以下，“ノード”と呼ぶ)へ移動させるための移動制御方式について述べる．提案する方式では，各ノードのコーディネータ MA 間で交渉処理を行い，サービス MA の最適な移動先を決定する．

4.1 交渉処理

4.1.1 交渉開始のトリガ

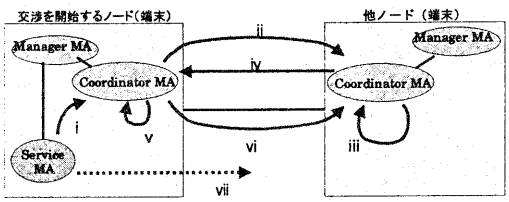
交渉処理は，監視 MA からの通知をトリガとして開始される．監視 MA は，あらかじめ定義された条件(移動条件)に基づいて，環境の変化をサービス MA

に通知する．環境変化の通知を受けたサービス MA は，自ノードのコーディネータ MA に交渉開始を要求する．環境変化の具体例を以下に示す．

- 利用可能な計算資源(CPU, メモリ, 帯域幅等)やバッテリ残量の減少等により，自ノードがサービス MA の動作条件を満たさなくなった場合．
- 他のエージェントとの協調が必要になった場合．
- ネットワークトポロジが変化した場合(新しいノードのネットワークへの参加等)．

4.1.2 交渉プロトコル

交渉処理で用いるプロトコルは，複数のエージェント間のタスク割当てに一般的に利用される“契約ネットプロトコル”[3]を基本とする．典型的な交渉プロトコルは，“タスク告示”，“入札”，“落札”の 3 段階の交渉を行う．交渉プロトコルの流れを図 3に示す．また，交渉中にノード間で交換されるメッセージの 1 例を図 4に示す．図 3の“協調エリア”とはタスク告示をマルチキャストする範囲を表している．



i. サービス MA は，評価基準と必要なパラメータを引数にしてコーディネータ MA に交渉開始を要求．
ii. コーディネータ MA は協調エリア内の各ノードに，移動先決定のための仕様をマルチキャスト．＜タスク告示＞
iii. 各ノードのコーディネータ MA は自ノードの状態を調査．
iv. 各ノードの情報が要求元のノードに返送．＜入札＞
v. 要求元のノードのコーディネータ MA は返送されたパラメータに従って評価式を計算し，移動先を決定．
vi. 決定した内容を各ノードに報告．＜落札＞
vii. 決定した移動先にサービス MA が移動．

図 3 交渉プロトコルの流れ

＜タスク告示＞

送信先アドレス	：エリア内の全ノード
送信元アドレス	：ノード A
メッセージの型	：タスク告示
番号	：△△△△
タスク概要	：サービス MA（X）の移動
資格仕様	：サービス MA（X）が実行可能であること
入札仕様	：利用可能なノード性能（仕様）P 利用可能なディスク容量（状態）S 他ノードとの間のネットワーク性能（状態）C バッテリー残量（状態）E
期限	：YYYY 年 MM 月 DD 日　hh:mm:ss

＜入札＞

送信先アドレス	：ノード A
送信元アドレス	：ノード B
メッセージの型	：入札
番号	：△△△△
契約者概要	：P = 1.5 S = 1 MByte C = 3 Mbps E = 30 min

＜落札＞

送信先アドレス	：エリア内の全ノード
送信元アドレス	：ノード A
メッセージの型	：落札
番号	：△△△△
タスク仕様	：移動先はノード B

図 4 交渉プロトコルのメッセージ例

交渉プロトコルの流れの中の処理(v)において，移動先を決定するための評価基準は，“評価式”と“制約条件”とで構成される．ここで，制約条件はサービス MA の移動先となるノードが満たすべき条件を示す．また，評価式は，ノードごとに処理時間や通信コスト等を算出するための式である．制約条件のもとで評価式の値を最小化あるいは最大化するノードを選択することにより，移動先を決定する．また，自ノードも評価の対象となるので，“移動しない”という決定も選択肢の中に自然な形で含まれている．

4.2 移動先端末の判定

ここでは，処理時間を最小化する場合を例に，移動先端末の判定方式を，次に示すモデルを用いて説明する．

4.2.1 サービス MA のモデル化

サービス MA は下記のような値を内部データとして保持する．

- 平均サービス時間：T_{task}
- 残余サービス時間：T_{rest}
- 移動時の平均データ長：L
- 平均データ通信量：D
- 移動時のオーバヘッド時間：T_{OH}
- 移動条件
- 制約条件
- 評価式

ここで，平均サービス時間 T_{task} はサービス MA の処

理量であり，実際に処理を行うノードの性能 P と処理時間 t の積で表される(T_{task} = P × t)．また，残余サービス時間は，交渉を行う時点でのサービス MA の残りの処理量を表している．移動時の平均データ長はサービス MA の移動時のサイズ(コード+データ)であり，平均データ通信量はサービス MA が他サービス MA と通信を行うときのデータ通信量である．

4.2.2 ノードのモデル化

ノードは他に負荷がない状態での性能 P_0(固定値)と，それ以外に下記の状態を持っており，これらの値は各ノードの監視 MA により収集される．

- 利用可能なノード性能：P
- 利用可能なディスク容量：S
- バッテリ残量：E
- ノード i, j 間のネットワーク性能：$C_{i,j}$

利用可能なノード性能とは，そのノードが移動先として選ばれた場合に，サービス MA が利用可能なノード性能の推定値である．

4.2.3 評価式

処理時間を最小化する場合の評価式 f は以下のようになる．

f = サービス MA の残りの処理時間
　　＋サービス MA の残りのデータ通信処理時間
　　＋サービス MA が移動する場合の移動時間
　　＋実行要求元ノードに戻るのに必要な移動時間　(1)

ここで，実行要求元ノードとはサービス MA の実行を要求したアプリケーションが存在するノードを指す．右辺の第 4 項は，サービス MA が処理完了時に実行要求元ノードに戻る必要がある場合に追加される．式(1)は，前述のサービス MA とノードのモデルを用いると，以下のように表せる．

$$f_i = T_{rest} \times 1/P_i + D \times 1/C_{i,j} + h(i, i_1) \times \{L / C_{i,i1} + T_{OH}\} + h(i, i_0) \times \{L / C_{i,i0} + T_{OH}\} \quad (2)$$

ただし，h(x, y) = 1(x≠y), 0(x=y)とする．また，制約条件は，例えば最低限必要なディスク容量とバッテリ残量をそれぞれ S_{min}, E_{min} として，

$$S_i > S_{min}$$
$$E_i > E_{min} \quad (3)$$

と定義する．ここで，iは協調エリア内のノードを表し，i_0はサービス MA の実行要求元ノードを，i_1は現在サービス MA が存在するノードを，jはサービス MA がデータ通信を行う相手ノードをそれぞれ表す．

交渉を開始したコーディネータ MA は，各ノードから収集した情報をもとに，制約条件を満たす各ノードの評価式の値を計算し，その値が最小となるノードを移動先として決定する．

4.3 動作例

ここでは，2 つのシナリオを例に，交渉処理の具体的な動作を示す．

4.3.1 シナリオ 1

図 5のような環境において，PDA を携帯したユーザが無線を用いてネットワークに参加し，ネットワーク上の計算機資源を利用して，PDA 上のアプリケーションの処理を実行する場合を検討する．

今，時刻 t_0〜t_5 に図 6に示す環境変化があったとする．このとき，提案方式では，時刻 t_0, t_2, t_3, t_4 に起きた環境変化(t_0 はサービス MA の実行開始，t_2 はノードの性能低下，t_3 はノードのバッテリ残量減少，t_4 は新しいノードの追加)をトリガとして，交渉を開始する．交渉の結果，サービス MA は C→A→B→D の順でノードを移動する(図 6の○印が付いたノードにサービス MA が存在する)．

図 7にサービス MA の処理完了までの時間経過を示す．図 7には，サービス MA がノード A から移動しない場合と，時刻 t_0 で最も性能の良いノードである C にサービス MA を固定した(交渉を行わない)場合の動作も併せて示す．図 7では，交渉にかかる時間(実線のグラフの水平になった部分)を含めても，交渉を行って最適なノードに移動させた方が，最終的にサービス MA の処理を早く完了できることが示されている．

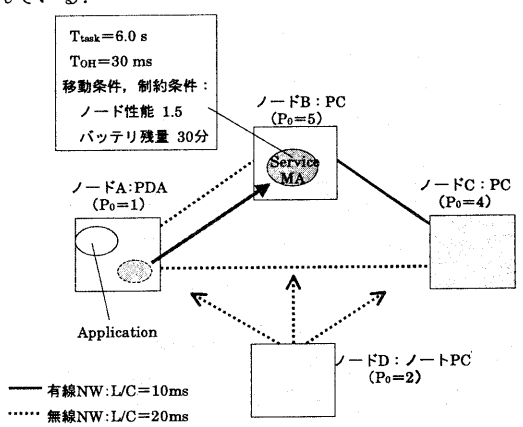


図 5　シナリオ 1 の構成図

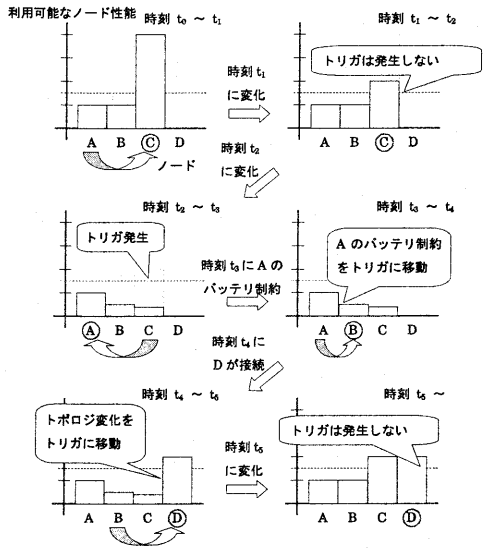


図 6　シナリオ1における環境変化

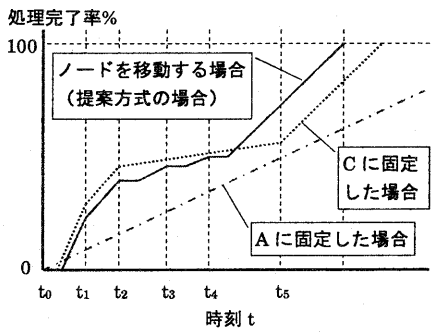


図 7　サービス MA の処理の時間経過

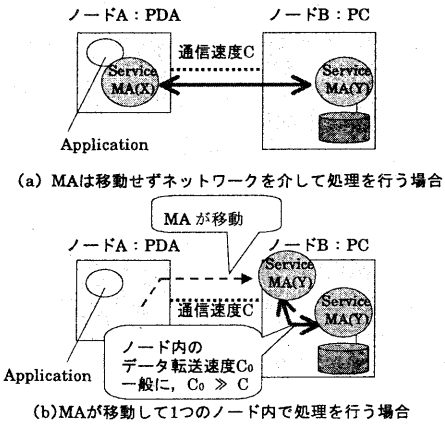


図 8　シナリオ2の構成図

4.3.2　シナリオ2

図 8(a)に示すように，PDA(ノード A)上のサービス MA が，他ノード(ノード B)上のエージェントと通信しながらノード B 上にあるデータの処理を行う場合を検討する．この例では，通信量が非常に多ければ相手ノードに移動して処理を行う方が効率的であると考えられる(図 8(b))．提案方式はこれを実現できることを示す．

まず，データ通信量に注目するため，ノード A，B の利用可能なノード性能は等しいと仮定すると，各ノードにおける処理時間も等しくなり，それを T とおく．今，ノード A にいるサービス MA(X)がノード B にいるサービス MA(Y)と協調する必要が生じたことを交渉開始のトリガとする．このときの評価式は次式で表せる．

$$f_A = T + D/C$$

$$f_B = T + D/C_0 + 2(L/C + T_{OH}) \quad (4)$$

移動を行うのは，$f_B < f_A$ の場合であり，さらに，$C_0 \gg C$ という条件を考慮すると，式(4)より次式に示す条件が導かれる．

$$D > 2(L + C \times T_{OH}) \quad (5)$$

この式は，データ通信量 D がある値より大きければ，移動を行うことを示しており，通信量が多ければ他ノードに移動して処理を行う方が効率的であるという考え方に一致している．また，この目安となる値は，エージェントが移動する際のデータ転送量と，移動時のオーバーヘッド時間内にネットワークを介して転送できる最大データ量の和の 2 倍(往復分)として表されており，上記のような仮定をおいた場合には妥当な値である．

5.　考察

モバイルエージェントは，分散環境におけるさまざまな課題に対して重要な解決法を提供するものとして，近年特に注目を浴びている[4]．モバイルエージェントにとって，“移動(migration)”は最も重要な特徴の 1 つである．本稿では，この“移動”を制御する方式に焦点を当てて議論を進めてきた．ここでは，交渉プロトコルを用いた移動制御方式について考察する．

本稿では，移動先を決定するための交渉プロトコルとして契約ネットプロトコルを用いている．契約ネットプロトコルと同様に，複数のエージェント間の協調プロトコルとして用いられるものに，オーク

ション(競売)がある．文献[5]では，複数の組織におけるリソース制御の問題を解決するために，オークションを採用している．オークションを用いたアプローチでは，複数の組織に共通の“通貨”という価値基準を持たせることによりリソース利用量のバランスを保つ．しかし，オークションを用いたアプローチを実現するためには，複雑な仕組みをシステムに実装する必要がある．また，システムを利用する端末(あるいは組織)間で，あらかじめ通貨の配分等の調整を行ってからでないと利用できない．本稿で想定するモバイル環境のように，リソースの限られた端末が接続するローカルなネットワーク内で，モバイルエージェントが移動する場合には，契約ネットプロトコルのような“実装が容易”で“導入に手間のかからない”プロトコルが適していると考える．

6. まとめ

本稿では，筆者らが提案した“モバイルエージェントを基盤とするミドルウェア”を実現するための具体的な移動制御方式について検討した．この移動制御方式は契約ネットプロトコルを基本とする交渉プロトコルによって他ノードの情報を収集し，モバイルエージェントの内部情報として定義された評価基準に従って収集された情報を評価し，移動先を決定する．そして，端末の性能や通信帯域に制約のあるモバイル端末を利用する 2 つのシナリオを例に，提案方式の具体的な動作を示した．提案方式は，実装や導入が容易という利点があり，特にリソースの限られた端末が接続するローカルなネットワーク内で，モバイルエージェントが移動する場合に適している．

今後の課題としては，提案方式を定量的に評価するために数値シミュレーションを行うことや，実際のエージェントシステムに実装し効果を検証することが挙げられる．

謝　辞

日頃ご指導頂く当社情報通信研究所 弓削 哲也 副所長および 藤井 輝也 担当部長に感謝いたします．

文　献

[1] K. Hiroshige, K. Kawakami, H. Sasaki, Y. Okataku, and S. Honiden, “Mobile Agent-Based Middleware for Mobile Terminals,” http://computer.org/dsonline/0107/features/hir0107.htm

[2] K. Geihs, “Middleware Challenges Ahead,” IEEE Computer, vol.34, no.6, pp.24-32, June 2001.

[3] R. G. Smith, “The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver,” IEEE Transactions on computers, vol.c-29, no.12, 1980.

[4] 本位田真一，飯島正，大須賀昭彦，“エージェント技術,” 共立出版, 1999.

[5] J. Bredin, D. Kots, and D. Rus, “Market-based Resource Control for Mobile Agents,” In Proc. of the Second International Conference on Autonomous Agents AA98, pp.197-204, Mineapolis, USA, May 1998.