モバイルエージェントのためのセキュリティ機能についての検討

永 井 保 夫*

モバイルエージェントは分散コンピューティングに対する新しいパラダイムを提供するものである。しかしながら、モバイルエージェントに関連するセキュリティの脅威に対しては十分な検討が必要である。ここでいうセキュリティの脅威とは悪意のあるエージェントだけでなく、悪意のあるホストにより引き起こされるものである。前者は、悪意のあるモバイルエージェントがホストを攻撃する場合に相当し、具体的には計算資源への不当なアクセスや計算資源の過剰な消費などが考えられる。後者は、移動先の悪意のあるなホストがモバイルエージェントを攻撃する場合に相当し、ホストによる攻撃に対して防御機能をもたないエージェントの情報への不当なアクセスや改竄（たとえば、エージェントへのホスト自身のタスクの組み込みやエージェントの状態の修正）などが考えられる。本論文では上記のような攻撃に対する保護機能をモバイルエージェントのセキュリティ機能とみなし取り上げることにする。以下では、まず、モバイルエージェントとその特徴について説明する。次に、モバイルエージェントのセキュリティとして必要とされる機能と代表的なシステムにおいて実現されている機能について報告する。

キーワード：モバイルエージェント，セキュリティ，ネットワーク，ソフトウエア，Java

Towards Security Functions for Mobile Agent Systems

Yasuo NAGAI

Mobile agent systems provide a new paradigm to distributed computing. Key elements of mobile agent systems on distributed environment are the security functions where the mobile agent systems can protect themselves against tampering by a malicious host（ex. code and execution integrity), the mobile agent systems can cancel the program they want to have executed（ex. code privacy), and the mobile agent systems can remotely sign a document without disclosing the user's privacy key（ex. computing with secrets in public). In this paper, we describe and discuss the security functions for mobile agent systems and survey the security functions of existing mobile agent systems.
keyword：mobile agent, security, network, software, Java

1. はじめに

モバイルエージェントは分散コンピューティングに対する新しいパラダイムを提供するものである［20］［4］［19］。しかしながら、モバイルエージェントに関連するセキュリティの脅威に対しては十分な検討が必要である。ここでいうセキュリティの脅威とは悪意のあるエージェントだけでなく、悪意のあるホストに

*東京情報大学総合情報学部情報システム学科
Tokyo University of Information Sciences, Faculty of Informatics, Department of Information Systems
2003年12月1日受理

より引き起こされるものである。前者の脅威は、悪意のあるモバイルエージェントがホストを攻撃する場合に相当し、具体的には計算資源への不当なアクセスや計算資源の過剰な消費などが考えられる。後者の脅威は、移動先の悪意のあるホストがモバイルエージェントを攻撃する場合に相当し、ホストによる攻撃に対して防御機能をもたないエージェントの情報への不当なアクセスや改竄（たとえば、エージェントへのホスト自身のタスクの組み込みやエージェントの状態の修正）などが考えられる。本論文では上記のような攻撃に対する保護機能をモバイルエージェントのセキュリティ機能とみなし取り上げる。以下では、まず、モバイルエージェントとその特徴について説明する。次に、モバイルエージェントのセキュリティとして必要とされる機能と代表的なシステムで実現されている機能について述べる。

2. モバイルエージェントの特徴

ソフトウエアエージェントは、ステーショナリエージェントとモバイルエージェントの2種類に分類できる。ステーショナリエージェントは、ネットワークマシンにより接続されたマシン間を移動する能力をもたず、マシン上に存在して処理をおこなうソフトウエアをあらわす。これに対して、モバイルエージェントは、図1に示すように、ネットワークに接続されたコンピュータ（エージェントシステム）間を移動する能力をもつソフトウエアをあらわす。モバイルエージェントは、自分自身で処理をおこなうだけでなく、ステーショナリエージェントと連携して処理をおこなう場合も多い。その場合には、従来おこなわれているステーショナリエージェント間で交渉や協調ではなく、モバイルエージェントとステーショナリエージェントとの交渉や協調がおこなわれる［10］。

モバイルエージェントを利用した場合の利点として、以下の7点があげられる。

・ネットワーク負荷の軽減
　エージェントの移動時のみ、ネットワークを利用することで、ネットワークのバンド幅を有効に利用できる。また、サーバでおこなっていた仕事の一部をクライアントに移動することが可能になる（負荷分散／機能分散）。

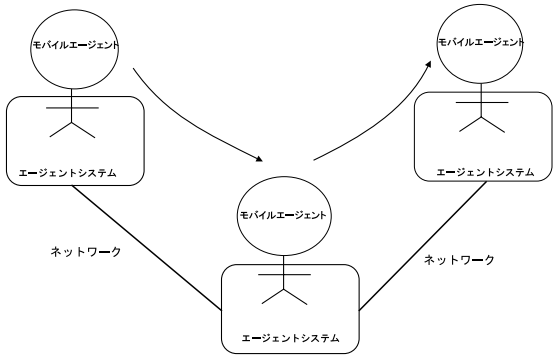

図1：モバイルエージェントシステムの構成

・リモートアクセスのローカルアクセスへの変換
　転送先のリソースへの直接制御ならびに直接アクセスが可能になる。
・プロトコルのカプセル化
　マシンの新たな性能要求などにより変更されていくプロトコルに追随してコードを改良していくことは容易ではない。モバイルエージェントを利用することで、このようなマシンの間で交わされるプロトコルの変更に基づく処理はモバイルエージェントの移動元と移動先のそれぞれの処理として実現することが可能になる。
・非同期かつ自律的な計算が可能
　モバイル機器に接続したままで作業をおこなう場合には、多くの通信コストが要求される。モバイルエージェントを利用した通信ではエージェントを送出した後、エージェントが処理を終了して戻ってくるまで通信回線を切断できるので、通信コストの抑制が可能である。
・動的な適合が可能
　モバイルエージェントは実行環境を認識し、環境の変化に対して自律的に対応できる。
・異種システムの自然なモデル化
　ネットワーク上での異なる計算機やソフトウエア環境上で動作する異種システムのモデル化では、モバイルエージェントの利用によりこれらのシステム間を柔軟につなぐことが可能となる。
・頑強でかつ誤りへの耐性力がある

予期できない状況やイベントに対する動的に応答するモバイルエージェントの能力を利用することで、頑

強でかつ誤りへの耐性力のある分散システムを構築することが容易化される。このように、モバイルエージェントはネットワークコンピューティングに対する強力で均一なパラダイムを提供している。

以下では、ネットワーク上でのアプリケーション開発に用いられるパラダイムとして、クライアントサーバによるパラダイム、コードオンデマンドによるパラダイム、モバイルエージェントによるパラダイムを取り上げ、それぞれの特徴を比較することで、モバイルエージェント利用の有効性を示す。

・クライアントサーバによるパラダイム［17］
クライアントサーバによるパラダイムでは、サーバマシンは資源（たとえば、データベース）をアクセスするサービスを提供する。サーバマシン上にはこのようなサービスを実現するコードが置かれており、そこにはサービスを実行可能な資源が割り当てられている。また、クライアントマシンは自分が利用するサービスがどのサーバマシンのコードにより提供されるかを知ることが必要となる。
・コードオンデマンドによるパラダイム［1］
コードオンデマンドによるパラダイムでは、要求されたときに必要なコードを得ることができるようになる。たとえば、マシンAでは必要なコードがないためにタスクが実行できず、一方、ネットワーク中の他のマシンBでは必要なコードが提供されているケースを考えてみる。このような場合には、マシンAがマシンBよりコードを受け取ると、ローカルな資源が提供されているマシンAによりコードの実行がおこなわれる。コードオンデマンドによるパラダイムにより、クライアントサーバによるパラダイムとは異なり、すべての必要なコードがダウンロードされるので、マシンAはネットワーク上の他のマシンが提供するサービスについて知る必要がなくなる。Java アプレットやサーブレットはこのようなコードオンデマンドによるパラダイムの代表例である。
・モバイルエージェントによるパラダイム
モバイルエージェントによるパラダイムでは、ネットワーク中のマシン上にあるサービスを提供するコードならびに資源を利用できるようになるので、高い柔軟性が得られる。たとえば、モバイルエージェントを移動させることで、移動先のローカルな資源を利用した処理が実現できるようになる。これにより、特定のマシンに置かれているコードを利用する代わりに、ネットワーク中の他のマシンのコードを利用できるので、サービスを柔軟に受けられるようになる。

3. モバイルエージェントのセキュリティ機能

ネットワーク上のセキュリティでは、次の 3 点を考えることが必要である［21］［2］。
・機密性
開示を許可していないデータの保護
・完全性
改竄破壊に対するデータの保護
・可用性
必要に応じたデータの利用

モバイルエージェントやエージェントシステム（モバイルエージェントが移動するプラットフォームをあらわし、システムによってはプレース、エージェンシー、アプリケーションなどと呼び方が異なる。また、上述したステーショナリエージェントを示すこともある）の実行により、ホスト上で提供されているサービスの妨害、データの権限なしアクセスや利用、データ変更や破壊、誤りを含んだデータの追加によりデータ修正や破壊などが発生する可能性がある。

このような状況を回避するためには、モバイルエージェントにおけるセキュリティ機能として、以下の 3 つのケースに対応することが必要である。

・不正なモバイルエージェントからの攻撃に対するホストの保護
不正なモバイルエージェントのホストへの攻撃を防御することを示している。具体的な攻撃としては計算資源への不当なアクセスや計算資源の過剰な消費が考えられる。
・ホストからの攻撃に対するモバイルエージェントの保護
モバイルエージェントを移動先の不正なホストの攻撃から防御することをあらわす。具体的な攻撃としてはエージェントのもつ情報への不当アクセス改竄（たとえば、エージェントへのホスト自身のタスクの組み込みやエージェントの状態の修正）が考えら

れる。
・他のエージェントからの攻撃に対するモバイルエージェントの保護
　他のモバイルエージェントやステーショナリエージェントからの攻撃をあらわす。

このようなモバイルエージェントのセキュリティ機能を詳細化すると、以下の項目を考慮する必要がある。

・モバイルエージェントの送り手、作成者、所有者の認証
　－誰がエージェントの責任をもつか？
　－誰がエージェントのコードの責任をもつか？
　－エージェントは不正に利用されていないか？
　ということを認証できなければならない。
・エージェントの認証
　不正なホストへ移動する前に、移動先のエージェントの素性を調べるために必要である。不正なホストからの攻撃を防御するために用いられる。
・エージェントシステム間での安全な通信
　データやコードが安全に通信されるためには、暗号化だけではなく、署名も必要である。これらを組み合わせることで、データの盗み読みだけでなく、改竄にも効果がある。
・リモートなエージェント生成のためのクライアントの認証
　非エージェントシステムではクライアントアプリケーションの認証機能が必要である。なお、クライアント認証では、クライアントの起動するエージェントに関する証明書に基づき、どのようなセキュリティポリシーが利用されるかを決定することが必要となる。
・エージェントシステムの相互認証
　人間の介在なしで動作するエージェントシステムはお互いに認証可能でなければならない。
・認証結果と証明書へのエージェントシステムのアクセス
　エージェントによる通信が発生した場合に、移動先であるエージェントシステムはエージェントと移動元であるエージェントシステムの証明書を取り出し、認証できなければならない。
・エージェント認証と委譲
　エージェントが目的先であるエージェントシステムへ移動する場合には、移動が成功すればエージェントの証明書はエージェントとともに移動しなければならない。
・エージェントとエージェントシステムのセキュリティポリシー
　エージェントとエージェントシステムは、以下の項目に対する動作の規定（この規定をセキュリティポリシーという）が要求される。
　－エージェントの能力に関する制限や許可の設定
　－計算資源に関する消費制限の設定
　－計算資源のアクセスに関する制限や許可の設定

4. 代表的なモバイルエージェントのセキュリティ機能

本節では、まず、モバイルエージェントではないが代表的なモバイルコードであり、多くのモバイルエージェントがベースとするJava言語のセキュリティ機能について説明する。

それから、代表的なモバイルエージェントとして、Voyager［5］、Odyssey［13］、Aglets［7］［10］、Kafuka［6］、Agent Tcl［9］、Ara［18］、Jumping Beans［8］、Plangent［12］［14］を取り上げ、これらの特徴とセキュリティ機能について紹介する。なお、Kafukaについてはマルチエージェントによるアプリケーション構築を目的にしているが、モバイルエージェントの実現機能も提供しているため、モバイルエージェントとみなして取り上げた。

1. Java

(a) JDK 1.1以前のセキュリティ機能［11］
・不正なメモリアクセスの防止
　変数やメソッド、さらにはシステムそのものへのアクセス行為を防止させるものである。具体的には、不正なデータ型変換の禁止、ポインタの排除、メモリ解放の管理、文字列や配列の大きさを超えた参照の禁止、クラスベリファイァによるバイトコード自体のチェックなどの手段があげられる。
・不正なクラスアクセスの防止
　サンドボックスはネットワークを介して呼び出されたクラスをクラスローダの管理下におき、あるクラスが他のクラスに勝手にアクセスできないよ

うにするメカニズムである。クラスローダは指定されたクラスのクラスファイルをディスクやファイルから読み出すために用いられる。

・不正なリソースアクセスの防止

セキュリティマネージャは不正なリソースアクセスを防止する機能である。セキュリティマネージャは、悪意をもって作成されたプログラムがJava仮想マシン（JVM)のシステム・クラスのメソッド呼び出しをシステムクラスに回避させ、破壊的な行為を防止する。このように、Java言語では、セキュリティマネージャは必ずシステムクラスに含まれる危険なメソッドの実行可否を確認する。

(b) JDK 1.1のセキュリティ機能［11］

信頼できるアプレットに対するセキュリティ上の制約を緩和する仕組みとして、電子署名付きアプレット機能が提供された。これにより、Javaプログラムをネットワークを介してダウンロードでき、ローカルなアプリケーションとして利用できるようになった。しかしながら、電子署名付きアプレットでは、危険なメソッド実行を許すか許さないかの二者選択であり、アクセス制御機能である認証と権限のうち、後者の権限については全く提供されていない。

(c) JDK 1.2のセキュリティ機能［3］

JDK 1.2のセキュリティ機能では、JDK 1.1の機能と比較して以下の点について改善されている。

・設定変更の容易性

JDK 1.2ではセキュリティマネージャの直接の改造なしに、セキュリティの設定が可能である。

・アクセス制御の安全性

詳細なレベルでのアクセス制御が可能となり、安全性の確認されている機能のみの実現を可能にする。

・クラス単位での制御

アプレット単位でのアクセス制御を提供していたJDK 1.1と異なり、JDK 1.2ではクラス単位でのアクセス制御が可能である。

・ローカル･ディスク上でのクラス制御

JDK 1.1では、システムクラスやその他のローカルディスク上のクラスに対する無条件なアクセスを許可していたが、JDK 1.2ではシステムクラス以外のすべてのクラスをアクセス制御の対象とできるようになった。

・電子署名なしクラスの制御

電子署名なしのクラスも、その保存場所さえ決定していれば、アクセス制御可能である。

2. Voyager

VoyagerはJava言語に基づいたエージェントによる分散コンピューティング用のプラットフォームである。Voyagerはオブジェクトメッセージング機能を提供する一方で、オブジェクトをエージェントとみなすことで、ネットワーク中を移動させる機能をもつ。VoyagerではJava言語に基づいたオブジェクトリクエストブローカー機能とモバイルエージェント機能とを結び付けることで、従来の分散プログラミング技術とエージェントベースの分散プログラミング技術の両方を用いたネットワークアプリケーションの作成が可能である。VoyagerはJavaオブジェクトに特化したORBシステムを有し、ORBを介したオブジェクト／エージェントの移動をおこなう。また、多様な遠隔メソッド呼び出し（RPC）がおこなえ、Common Object Request Broker Architecture（CORBA）／Internet Inter-ORB Protocol（IIOP)、Distributed Component Object Model（DCOM）などとの親和性がよいことが特徴である。

Voyagerのセキュリティ機能では、JavaのSecutiry Managerを継承したVoyagerSecurityManagerを作成し、これを利用することでモバイルエージェントの実行可能な操作を制限できる。このSecurityManagerのコードを修正することでユーザのセキュリティのニーズを組み込むことが可能である。

3. Odyssey

OdysseyはTelescriptをJava言語環境上で再実装したシステムである。オブジェクトの移動にはRemote Method Invocation（RMI）／IIOP／DCOMが利用されており、ネーミングにはRMIが用いられている。Odysseyには、独自のローダ／インタプリタが提供されており、ユーザの記述したクラスはセットアップファイルを読み込ませて実行される。開発者は提供されているJavaクラスライブラリを利用して、モバイルエージェントアプリケーションを作成できる。Odysseyのセキュリティ機能としては、独自のSecurityManagerは提供されておらず、Telescriptで実現されてい

たモバイルエージェントやエージェントシステムの認証機能やリソースに関するアクセスコントロール機能が提供されている。

4. Aglets

Agletsはモバイルエージェント構築環境として、以下の機能を提供している。

・ステーショナリエージェント（オブジェクト）とモバイルエージェント（オブジェクト）の実現
・非同期的な処理および同期的な処理
・ローカルオブジェクトおよびリモートオブジェクトの操作
・ネットワーク接続時と非接続時の動作

AgletsはJavaアプレットのコンセプトをモバイルエージェントに反映させたものであり、移動先はURLを利用する。上記のOdyssayと比較すると、Agletsは多くのエージェントの特徴である自律性を有しておらず、移動可能なオブジェクト（Mobile Object）とみなすことができる。

セキュリティ機能は、独自のAgletSecurityManagerにより実現されており、ユーザの要求に応じたカスタマイズが容易におこなえない。Agletsでは固有のセキュリティモデルを提案しており、具体的にはローカルリソースのアクセスコントロール機能とユーザやグループの権限（オーソリティー）の定義・チェック機能が実現されている。また、Agletsはユーザが実行モニタから生成したAgletsをTrusted Agletsとみなし、Trusted Agletsから生成されたAgletsもTrusted Agletsとしている。一方、外部から送り込まれたAgletsとこれらのAgletsが生成するAgletsは、Untrusted Agletsとして区別され、これらの区別毎に、ファイル、通信ポート、その他のリソースへのアクセスを設定することが可能である。

5. Kafuka

KafukaはJava言語環境を利用した、マルチエージェントによる分散アプリケーション構築のためのライブラリである。エージェントはJavaのRMI（Remote Method Invocation）をベースとしている。Kafukaの特徴としては、リフレクション機能、リモートエバリュエーション、分散ネームサーバ、モバイルエージェント、カスタマイズ容易なセキュリティモデルなどが挙げられる。上記のセキュリティ機能では、あらかじめ決められたコードしか実行しないオブジェクトを比較して、より細かなセキュリティ機構が実現されている。このセキュリティ機構は、次の3階層のアクセスコントロールにより実現されている。

・システム･リソースへのアクセス制限
JavaのSecurityManagerをそのまま利用する。
・属性毎のアクセスモードの設定
エージェントの内部状態変数やアクション定義をあらわす属性をエージェントの外部に公開するか否かを設定する。
・Java言語を用いたアクセス制御
アクセスをおこなったエージェントのIDや属性、現在時刻、内部状態などを組み合わせて、アクセス権を動的変化させることができる。これにより、アクセス権をユーザがカスタマイズ可能になる。

6. AgentTCL

AgentTCLはスクリプト言語Tclにより記述されたモバイルエージェント構築言語である。AgentTCLではナビゲーション＆通信サービス、セキュリティメカニズム、デバッグ＆トラックツールが提供されている。AgentTCLは移動するエージェントと、局所変数とインストラクションポインタを含んだ全体の実行状態を移動可能な機能をもつサーバから構成される。エージェントが新しいマシンに移動する場合には、目的先マシン上のサーバにエージェントの全体状態を送信する。目的先サーバではTclの実行を開始し、この実行環境上での状態情報のロードにより、エージェントの終了時点からエージェントを再実行できる。AgentTCLのセキュリティとして、以下の機能が提供されている。

・エージェントのプライバシーを維持するために、マシン間で送信されたエージェントやメッセージを暗号化する機能
・新しいホスト対するエージェント認証用に、マシン間で送信されたエージェントやメッセージを電子署名化する機能

・システムリソースへのアクセスを管理するリソースマネージャ機能
・信頼できるコードを解釈実行するインタープリタと信頼できないコードを解釈実行するインタープリタとの分離

7. Ara（Agents for Remote Action）

Araはスクリプト言語Tclを利用したポータブルなモバイルエージェントで、異質なネットワーク環境上で安全な動作が保証されるプラットフォームを提供する。Araでのモバイルエージェントでは、安全でかつポータブルな実行をおこなう機構をサポートしている。Araの基本的なセキュリティ機能として、インタープリタでのメモリ保護機構が提供されている。さらに、これ以外にはファイル、CPU時間、メモリー、ディスク容量などのリソースに関するアクセス制御（アクセスコントロール）、エージェントの移動先での許容権についての設定などの機能が提供されている。

8. JumpingBeans

JumpingBeansは、ネットワーク上を移動するJavaアプリケーションを実現するフレームワークであり、主にネットワークデバイス管理に適用されている。JumpingBeansではアプリケーションは実行時の状態を保持したまま移動可能であり、クライアント側の実行環境がすべてJava言語により記述される。システム全体のメモリ消費量はORBを含み130Kバイト以下であり、Personal Javaに準拠している。

JumpingBeansのセキュリティ機能としては、以下が提供されている。

・アクセスコントロールリストACL（Access Control List）によるリソースのアクセス制御
・公開鍵／秘密鍵と証明書の利用による認証
・Javaセキュリティ機能を利用したデジタル署名の実現
・検査ログの生成・利用によるモバイルエージェントの監視

9. Plangent

Plangentの名前は、Planning Agentから取ったもので、各エージェントがプランニングという推論機構を持ち、ネットワーク上を自律的に移動することが大きな特徴となっている。Plangentにおけるエージェントは、人間の頭脳に相当するしくみとしてプランニング機構を、手に相当するしくみとしてプラン実行機構を、足に相当するしくみとしてネットワーク移動機構を持っている。このエージェントがユーザからの要求を受けとると、以下のように行動する。

(a) ユーザからの要求を受けとったエージェントは、プランニングによって、どこで何をするかといった自分の行動計画（プラン）を立てる。
(b) その行動計画に基づいて、必要な情報やサービスのある場所まで移動する。
(c) 移動先の情報やサービスを活用して計画を実行する。
(d) 予期せぬ事態によって計画の実行が失敗した場合、再プランニングによって、状況に応じた行動計画を作りなおす。
(e) 目標が達成されるまで行動計画の作成と実行を繰り返す。
(f) 最後にユーザのところへ戻って結果を報告する。

Plangentにおけるセキュリティ機能としては、信頼度という概念を導入することで、エージェントの信用度、移動先のノード（マシン）の信用度、プランニングを行う場としてのノードの信頼度を定義し、これに基づいたセキュリティ機能を実現している。

5. まとめ

本論文では、モバイルエージェントを利用したシステムを実用化していく上で不可欠な機能であるセキュリティ機能について説明した。まず、セキュリティ機能の実現に必要と考えられる項目を示すとともに、代表的なモバイルエージェントのセキュリティ機能の概要を述べた。これからの課題として、人間の代理人としてのエージェントの信頼度をどのように決定していくか、さらにエージェント自体のプライバシーについてもどのように確立していくかを明確にしていく必要がある。

参考文献

[1] Antonio Carzaniga, Gian Pietro Picco, and Giovanni Vigna, Designing Distributed Applications with

Mobile Code Paradigms, Procs. of ICSE'97, pp.22-32, (1997).
[2] William M. Farmer, Joshua D. Guttman, and Vipin Swarup, Security for Mobile Agents: Issues and Requirements, The 19th National Information Systems Security Conference, pp.591-597, (1996).
[3] Li Geng, Inside Java2 Platform Security, Addidon-Wesley, (1999).
[4] GMD FOKUS and IBM, Mobile Agent System Interoperability Facilities Specifications, OMG TC Document orbos/97-10-05, November, (1997).
[5] G. Glass, Objectspace Voyager Core Package Technical Overview, White Paper, http://www.objectspace.com.
[6] Kafka Beta 1.3, Fujitu Laboratory, http://jp.fujitu.com.
[7] Guenter Karjoth, Danny B. Lange and Mitsuru Oshima, A Security Model for Aglets, IEEE Internet Computing, July-August, pp.68-77, (1997).
[8] Jumping Beans Whitepaper, http://www.jumpingbeans.com.
[9] David Kotz, Rober Gray, Saurab Nog, Daniela Rus, Sumit Chawla, and George Cybenko, Agent Tcl: Targeting the Needs of Mobile Computers, IEEE Internet Computing, July-August, pp.58-67, (1997).
[10] Danny Lange, Mobile Objects and Mobile Agents: The Future of Distributed Computing?, The European Conference on Object-Oriented Programming (ECOOP 98) Brussels, Belgium, July (1998).
[11] Gray McGraw and Edward Felten, Java Security, Hostile Applets, Holes, and Antidotes, John Wiley Sons, Inc., (1997).
[12] 永井保夫他，Plangent I　インテリジェント・ネットワークエージェント，人工知能学会ホットトピックスと並列人工知能研究会SIG HOT/PPAI-9602-6，pp.29-36，(1996).
[13] Odyssey: Beta Release 1.0, http://www.genmagic.com/agents/.
[14] Akihiko Ohsuga, Yasuo Nagai, et. al, PLANGENT: An Approach to Making Mobile Agents Intelligent, Internet Computing Vol. 1, No. 4, July-August, pp.50-57, (1997).
[15] 小野康一，悪意のあるホストによる攻撃から移動エージェントをいかに守るか?，The Second Workshop on Internet Technology (WIT'99) (1999).
[16] Robert Orfali, Dan Harkey, and Jeri Edwards, Instant CORBA, John Wiley & Sons, INC., (1997).
[17] Robert Orfali and Dan Harkey, Client/Server Programming with Java and CORBA, second edition, John Wiley and Sons, New York, (1998).
[18] H. Peine, Security Concepts and Implementation for the Ara Mobile Agent System, The Seventh IEEE Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises, pp.236-242, (1998).
[19] Kurt Rothermel and Radu Popescu-Zeletin (Eds.), Mobile Agents, First International Workshop, LNCS-1219, Springer-Verlag, (1997).
[20] ソフトウエア科学会，チュートリアル「モバイルエージェント」，講習会資料シリーズNo.20, ISSN 1341-8718-20, (1999).
[21] Giovanni Vigna (Ed.), Mobile Agents and Security, LNCS 1419, Springer-Verlag, (1998).