マルチメディア通信と分散処理 85－24
（１９９７．１１．６）

実世界指向分散エージェントシステムアーキテクチャ(CASA)における保守運用管理の検討

岡本 克也　　寺本 昌弘　　庭野 栄一　　久野 浩

NTT情報通信研究所

近年のインターネットの発展に見られるように個人を主体とした分散動的環境下での利便性の高い情報流通の枠組みへの要望がますます強くなっている。そのような課題に対する１つの解決策として筆者等は、実世界での個人／組織の社会的役割、ポリシーに着目した実世界指向分散エージェントシステムアーキテクチャ(CASA)を提案した。本論文では、CASAに基づく、エージェントサービスの開発／実行プラットフォームにおける保守運用管理について、CASAの特性を反映した要求条件、設計／実装方針を述べる。

Study on Operation, Administration and Management of Real-world-based Distributed Agent System Architecture(CASA)

Katsuya OKAMOTO, Masahiro TERAMOTO, Eikazu NIWANO, Hiroshi KUNO
NTT Information and Communication Systems Laboratories

The internet-related technologies have established world wide community with distributed huge amount of contents. They brought us much convenience while it is difficult for us to find ones we want efficiently. As an approach to make that situation better, we have proposed Common Agent System Architecture(CASA), which allows us to set up and provide multimedia services easily in distributed computing environment by personal software agents with role, policy and relationship on the real-world basis. In this paper, the requirements needed for Operation, Administration and Management(OA&M) of a software platform on which agents are handled and managed based on CASA characteristics and basic idea for its design and implementation are described.

1　はじめに

近年、インターネットのような柔軟かつ自律分散的なサービス提供、情報発信環境は情報の洪水をもたらし、所望のサービス、情報の特定が困難となる問題を引き起こしている。

これを解決する１つの手段としてエージェントと呼ばれる技術が研究開発されている[1],[2]。

筆者等は、実世界と同一視可能な1)個人の社会的役割に基づいたエージェントの機能化、2)実社会構造に基づいたエージェント社会の構造化、を指向する実世界指向分散エージェントシステムアーキテクチャCASA(Common Agent System Architecture)を提案した[3]。その研究の目的は、分散環境においてエージェントを利用・提供するためのエージェント社会基盤モデルの確立にある。筆者らは、CASAのエージェントが持つ特性を実現し、さらにコンピュータ上での保守運用管理機能を加えたCASAプラットフォーム(CAMP: Common Agent Management Platform)の実装を分散オブジェクト技術の１つであるCORBA[4]を適用して進めている。

本論文では、CAMPのオープンな運用管理の中で保守運用管理の実現指針を提案する。その中で、CASAの保守運用機能の特徴である；

・対象がオーナの役割とポリシーを持ったエージェント
・個人環境／社会環境による管理
・エージェントの活動環境としての堅牢性

を活かしたCAMP機能について述べる。

以降、２章でCASA、CAMPの概要、３章でCAMPでの保守運用管理の要求条件と実現指針を述べ、４章で現在の実装について簡単に触れる。

2　CASAについて

2.1　CASAアーキテクチャ

CASAは, CORBA, DCOM, JavaBeans, JavaORB, HORBなどの分散オブジェクト管理技術とエージ

ェントアプリケーションとの中間レイヤを規定する分散エージェントシステムアーキテクチャであり、様々なエージェント要素技術および資源の統合を狙いとする。また、CASAでは、利用者のもつ実世界モデルと同一視可能な体系的・普遍的なアーキテクチャの確立を狙う。

2.2 エージェントワールドモデル

本節では、CASAのエージェントワールド(AW)を構成するエージェント間の相互関係（構造）と役割（機能）について述べる。

エージェントとは個人／組織(以下、オーナ)の分身であり、オーナの役割とポリシーを反映した社会的行為の主体である。仮想社会は、社会的行為の主体(エージェント)の集合体である。以下にAWの構成要素を示す。

(1)サービスエージェント(Service Agent:SA)：SAはオーナの役割とポリシーを反映して他者にサービスを提供する。

(2)ユーザエージェント(User Agent:UA)：UAはオーナのポリシーを反映し、他者SAのサービス消費、オーナ支援を行う。

(3)自己管理エージェント(self-Administration Agent:AA)：AAは個人世界(Personal Agent World:PAW)のUA/SAへのポリシーの反映、実世界と仮想世界の同期などPAWの管理を行う。

制度・文化を共有するPAWの集合単位をSAW(Social Agent World)と呼び、PAWは少なくとも1つのSAWに帰属する。

また、SAは、SAWにおいて地理的関係や組織的関係に基づき各SA間で帰属関係を有する。

2.3 エージェントモデル

CASAにおけるエージェント構造を図1に示す。内から第3層までを必須特性とし、第4層はエージェントの能力に関するものである。

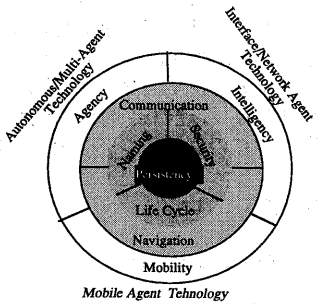

図1 CASAエージェント構造

エージェントは、オブジェクトであるが、AWモデルでの役割、ポリシー等を属性として有し、ポリシーに基づき、サービスを提供／利用するためのプロシージャを持つオブジェクトである。

2.4 エージェントプラットフォームモデル

CASAのエージェントはバックグランドで任意のオブジェクトを実行できる。各AWのサービス利用者には後者のオブジェクト群（オブジェクトワールド:OW）のオブジェクトは見えない(図2)。

しかし、AWとOWのオブジェクトが連携して動作するため、CASAプラットホーム（CAMP）では双方のオブジェクトが管理の対象となる。AWのオブジェクトおよOWのオブジェクトに対する管理機能単位を、それぞれ共通エージェント管理（CAM）および共通オブジェクト管理（COM）と呼ぶ（図3）。

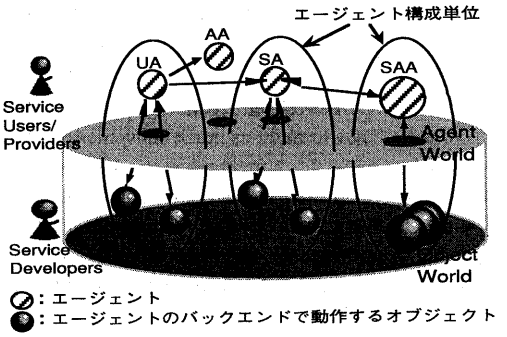

図2 CASAのエージェントモデル

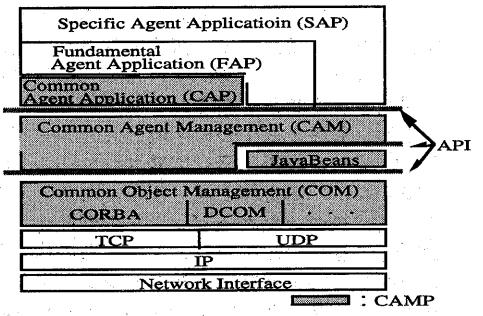

図3 システムアーキテクチャ

上記では、CAMPの管理をAWとOWに対応づけて分類したが、機能の観点ではエージェントのサービスに関する管理を行うサービス管理と、サービスの保守運用に関する保守運用管理に分

類できる（図4）。これらが相互に補完しながらエージェントシステムを運用・管理していく。

第3章では、CASAの特徴を踏まえた保守運用管理の管理方針と要求条件について述べる。

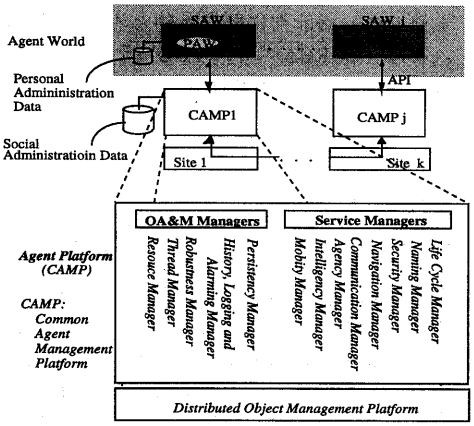

図4 プラットフォーム、及びシステムアーキテクチャ

3 CAMPにおける保守・運用管理

3.1 外部動向と全体方針

3.1.1 外部動向

エージェントに関する標準化組織としては、The Agent Society、Active Group、Foundation for Intelligent Physical Agents、IETF、Object Management Group、WWW Consortiumがあり、このうち、エージェント管理については以下の機関で具体的な検討がなされているものの、いずれもサービス管理に係わるものであり保守運用管理について統一的な検討はされていない。

●Foundation for Intelligent Physical Agents

エージェントが展開されるインフラを提供するエージェントプラットフォームの参照モデルの中で、プラットフォームのオペレーションの管理責任を有し、エージェントの生成, 削除,中断, 再開, 認証, 移動を管理するエージェントAMSが検討されている。

●Object Management Group

MobileAgentFacilityとして、与えられたクラス名からのエージェントの生成、エージェントの実行スレッドの中断／再開、エージェントの終了などが検討されている。

一方、CORBA、COM/DCOM、JAVA、Active Xなどの分散オブジェクト管理技術やコンポーネントソフトウェア技術では、オブジェクトの永続化やスレッド管理等の保守運用に係わる機能が部分的に規定／提供されている。

3.1.2 全体方針

本節ではCAMPの保守運用管理の総括的な実現方針について述べる。

(1)管理単位

CASAは実世界指向のアーキテクチャであり、個々のオーナの役割とポリシーを反映したエージェントから成るPAWの集合体によってAWが構成され、制度や文化を共有するPAWの集合としてSAWが存在する。このコンセプトに対応して、CAMPでのエージェント管理もPAWを基本単位と考えることとする。但し、個人社会を跨った事項、グループ内の個人社会で共通的な事項あるいはグループ内で一元性が求められる事項についてはSAWを管理単位とする。

(2)管理方針

PAWあるいはSAWのオーナは、それぞれ自分のポリシーによって個人社会とグループ社会を構築する。従って、エージェントの保守運用管理についても対応する個人社会やグループ社会のオーナの管理方針が反映される必要がある。具体的には、個々の保守運用管理機能を適用する契機や条件の選定がオーナの方針を反映することができ、またその選定（判断）のための材料をCAMPがオーナに提供できる仕組みを備えることとする。

(3)分散環境での保守運用管理

オブジェクトの特殊クラスであるエージェント、およびエージェントと動的に連携して動作するオブジェクトが、異なるホストに跨って実現されることを可能とする。従って、保守・運用管理機能は、分散環境にあるこれのオブジェクトを一体的に管理できる必要がある。

(4)実装の柔軟性を考慮した実現方針

分散オブジェクト管理技術やコンポーネントソフトウェア技術において、オブジェクトレベルでの保守運用管理に係わる機能が部分的に規定あるいは提供されているものの、汎用的である反面、使用方法は複雑でアプリケーション開発者に高度な利用技術が求められる。CAMPでは、まとまった保守運用管理機能をシンプルな利用方法で　提供することによって、エージェントおよび構成オブジェクトの開発者の利便性向上を図る。

一方、各種の分散オブジェクト管理等の技術は今後拡張されていくことが予想され、その場

合に当該技術の変化（拡張）によってCAMPの実装が大きく影響を受けることは避けるべきである。従って、保守運用管理の機能を個々の分散オブジェクト管理等の技術に依存する機能と独立な機能に分離し、CAMPの実装の柔軟性を確保することが重要である。

これらの点を考慮して、以下の方針でCAM層およびCOM層を実現する。

【機能分担】
＜CAM＞
・ひとつあるいは複数の分散オブジェクト管理技術が提供する汎用的な機能を統一された条件とフローで使用する制御機能、分散オブジェクト管理技術で提供されないエージェント固有の管理機能、およびそれらの機能の実行条件セットアップ機能（例えばエージェント間通信の一時的なブロック等）を提供する。
＜COM＞
・各種の分散オブジェクト管理技術およびソフトウェアコンポーネント技術が提供する機能やインタフェースの差異を隠蔽してCAM層に見せる機能を提供する。

【インタフェース】
＜CAM＞
・エージェントアプリケーションの開発者にとって、最終的な保守運用サービスの要求を曖昧性（多様性）の少ないシンプルなインタフェースを提供する。複雑になるほど、CAM層の存在価値は薄まる。
・背景となる技術の追加／変更に対して不変性が要求される。
＜COM＞
・基本的にはCAM層の開発者が使用するインタフェースであり、バックエンドのオブジェクト管理技術の差異を隠蔽したインタフェースを提供する。
・背景となる技術の追加／変更に対して可能な限り不変性を保つ必要がある。

(5)CAMP実装間のインタフェース

管理スコープの観点から、各々のSAWは異なるCAMPの実装上に構築されるべきであるが、例えばエージェントのリカバリなどSAW間に跨って保守運用管理機能が連携する必要のある場合、CAMP実装間でのインタワークが問題となる。CAMPがエージェントに対して見せる上記のCAMインタフェースがCAMP実装間のインタフェースにも統一的に適用されることが望ましいが、CAMPの実装間で保守運用管理インタフェースが異なる場合は、一方あるいは両方のCAMPでインタフェース変換機能の実装が必要である。

3.2 個別機能の要求条件と実現方針

本節では、機能別に要求条件と実装方針を示す。また、CORBAベースでCOMを実装する場合の考慮点についても述べる。

3.2.1 永続性

【要求条件】

AWを構成するエージェントにおける「Persistency（永続性）」とは、CASAエージェント構造における基本機能（図1）であり、エージェントの生成～消滅までの過程における様々な時点でのエージェントの保持するデータを２次記憶等に保存し、かつそのデータを必要時に再現することを意味する。特に構成オブジェクト群一体での保存は、動的に構成の変わるエージェントにとっては必須である。

○要求機能：
・管理単位：図5に示すようにUA、SA、AAの集合体としてのPAW単位
・保存／再現単位：エージェント、および構成オブジェクト群の整合をとり、保存／再現
・PAWのオーナのポリシー反映：
（保存契機）
・オーナの指定したタイミング
・エージェント破壊時
・エージェント消滅（Process Termination）時
（再現契機）
・オーナの指定時点
・エージェント起動時
・エージェントでの／への保存トリガにより、エージェントはSuspendされ、以後の要求を、ブロックすることで、保存データの整合性を保証

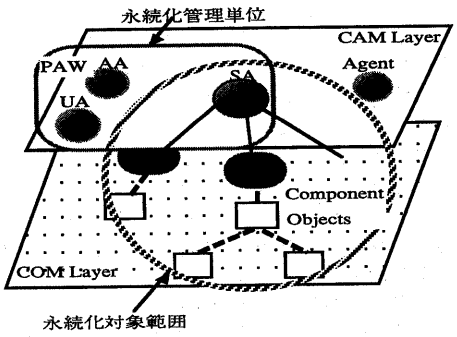

図5　永続化管理対象

【実現方針】

エージェントが上述の保存契機の主体となり，社会構造上、関係のある他エージェント、あるいは、所有関係にある全オブジェクトへ保存トリガをかけ、それに応じ、エージェントはSuspendされ、新規の要求をブロックする。

また、保存のトリガにより、構成オブジェクトの管理情報を用い、全構成オブジェクトの保存を行い、PAW単位で管理する。また、構成オブジェクトの動的な追加／削除などがあるのでその所有/構成情報は随時更新する。

COM層では、CORBA共通サービスのPSS(Persistent State Service Ver.2.0), Externalization ServiceやRelationship Serviceなどの採用も考慮する。

3.2.2 ログ＆アラーム管理

【要求条件】

CASAのエージェント間の通信は、PAWを跨ったUA-SA間通信である。従って、CAMPにおける障害発生の検出および解析を支援する本機能は、SAWで一元的に実現される必要がある。具体的には、通信トランザクションや各種エラーの収集機能、日時やエージェント等を条件とする検索機能、エラーログを契機に管理エージェント等にアラームを通知する機能が求められる。また、SAWが分散環境で構築される可能性があること、および不具合の発生に備えたログ収集の負荷がサービス用の通信に与える影響を低減すべきであることから、本機能もSAWの構成に応じて分散して実現する必要がある(図６)。

また、PAW単位で多数のエージェントが動作する環境では、統一的なログ管理機能は、ログでの障害解析等を円滑に行う上で重要となる。

○ログ出力イベント：

・Transaction Event：PAW間通信ログ
・Error Event：PAW内特定エージェントのError
・Alarm Event：PAW共通のクリティカルなError
・Periodic Event：リソース状態等を定期取得

アラームに関しては、ログ出力イベントとPAWとの関わり、およびSAの帰属関係を考慮して以下の対象に通知する必要がある。

・Error Event:該当エージェント, AA(オーナ), 上位SA(該当エージェントがSAの場合)
・Alarm Event:該当エージェント、SAA

【実現方針】

CAM層では、SAAにより、SAW単位にログ管理を行う。SAAは、ログ出力／参照の管理・制御、ログファイル管理、ログレコード検索、エラー通知機能を提供し、エージェントのオーナのポリシーにより、柔軟に制御可能とする。

COM層では、アラームをSAAが各PAWに通知する場合、CORBAのEvent Serviceを考慮する。

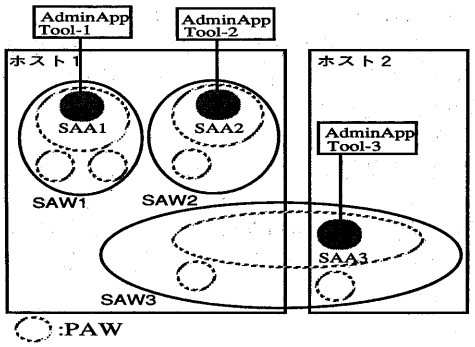

図6 ログ管理方針

3.2.3 履歴管理

【要求条件】

履歴管理は、各エージェントの行ったActionの詳細情報（日時、アクセス先、Operation、等）を管理する。履歴情報は課金や、制度・文化を共有する社会での動向把握に利用されることから、SAW単位で一元的に管理/保証することが重要となる。 履歴情報の分析により、SAWの特徴的な動向を抽出し、UAには、個人の趣向／ポリシーを反映したサービス性向上、SAにには、サービス戦略情報を提供する。

【実現方針】

SAWでのログ管理／制御と同等の機能を基本機能として提供する。

収集された交信／行動履歴情報は、マイニング[9], [10]等の統計処理を施し、有益情報（トレンド等）をUA/SAにリコメンドする[11]。

COM層では、SAAが各PAWに動向等の情報を提供する場合,CORBAのEventServiceを考慮する.

3.2.4 同時並行アクセス

【要求条件】

エージェントは、複数のオブジェクトで構成され、エージェントが要求を受付け、各オブジェクトに処理を依頼する形態となるため、複数の要求を同時に、各要求毎のセッション管理も行い処理する必要がある。性能的な面を考慮し,PAWのポリシーにより、スレッド数を動的に制御可能とする。

また、内部属性／DBに更新を行うような処理においては、アクセス時のロッキング機構が重要となる。

【実現方針】

PAW間でのエージェントへのアクセスは、エージェント同士で行われるので、CAM層では同時並行アクセス、セッション管理／スレッド管理機構を提供する。

また、COM層では、オブジェクト群が、複数の要求を処理し、データ等の更新処理を行うことから、内部データ不整合回避（排他制御）機構を提供する。本機能は、CORBA共通サービスのConcurrency Control Serviceの動向を踏まえ検討を進める。

3.2.5 リソース管理

【要求条件】

CPU/メモリ/プロセス等のリソースは、複数のPAWで分割使用されるため、リソースの監視、制御は、SAWでの管理が必要である。CASAでは、エージェントベースでの通信／処理が基本となることから、セッション/タスク管理に優先度を設け、エージェント間タスクを優先する。

分散環境で、PAWのポリシーのみでサービスが提供されるとシステム全体のリソースを圧迫する恐れがあるため、PAW毎にホスト別のリソース使用状況を管理できる必要がある。

【実現方針】

CAM層は、各エージェントの同時並行アクセスをサポートするスレッド取得／解放等のスレッド管理を行う。OS レベルでのリソース監視と連動し、スレッド生成の制御も提供する。

COM層では、CORBA製品の各プロセスへのアクセス監視、不要プロセスの消滅等の管理機能，およびCORBAシステム管理ファシリティを考慮する。

3.2.6 可用性

【要求条件】

多くのエージェントが相互に関連しながら処理を行うため、障害等によりエージェントがダウンした場合の各エージェント間／DB等との同期をとったリカバリは課金も絡むため非常に重要な要件となる。

・永続性による整合データの取得
・SAW単位で、関連するエージェントを対象
・PAWでのリカバリポリシーの反映(復旧時間、データ整合性を必要とする構成オブジェクト)
・重要内部データの２次記憶への保存
・各エージェントの交信履歴に基づく状態同期

【実現方針】

CAM層では、エージェント間、エージェントーオブジェクト間の関係性、および本機能の適用方針を考慮したPAW毎のポリシーでエージェントの堅牢性実現機能を提供する。

エージェント間／DBとの間でデータの整合性を必要とする場合、メッセージの交信履歴、および内部データのコミットファイルを用いた復旧手段を提供する。

COM層では、永続性、エージェントのコピー/移動、システム管理の基本機能として、CORBAのPSS, Externalization, LifeCycle, システム管理ファシリティの利用が考えられる。

4．おわりに

本論文では、CASAに基づく保守運用管理の機能について、要求条件、設計／実装方針を示した。現在、COM層のバックエンド機能としてCORBA(Orbix)を使用し、UNIX上で永続性、ログ管理機能から順次実装を進めている。永続性は、プリプロセッサ方式により静的な内部構成を持つ複合オブジェクトの永続化を実現した。さらに構成が動的に変化する場合の検討[8]を進める予定である。今後も、標準化動向、製品／技術動向を踏まえながらCASAの特徴を活かしたプラットフォームの実装を進めていきたい。

参考文献

[1]木下　哲男,"エージェントテクノロジの応用とその課題",信学技法A195-40,PRU95-155,1995-11

[2]Jeffrey M. Bradshaw,"Software Agents",The MIT Press, 1997

[3]庭野,千田,藤原 他, "CASA:実世界指向分散エージェントシステムアーキテクチャ",信学会,ソフトウェアエージェントとその応用シンポジウム講演論文集,'97.9

[4]http://www.omg.org/

[5]Usama Fayyad, Gregory Piatetsky-Shapiro, and Padhraic Smyth, "The KDD Process for Extracting Useful Knowledge from Volumes of Data," Comm. of the ACM, 1996.

[6]寺野　隆雄, "KDDツールの動向と課題," 人工知能学会誌, vol. 12, no. 4, pp.521-527, 1997.

[7]Paul Resnick,Hal R. Varian,"Recommender Systems",Comm. of the ACM,Vol.40,No.3,1997

[8]A.Biliris,S.Dar,N.H.Gehani,"Making C++ Objects Persistent:the Hidden Pointers",Software,Vol.23(12),1993