MobileAgentによる高機能情報サービスプラットフォームの提案*

6G－8

藤波 香織　　長岡 亨
NTTコミュニケーションウェア株式会社

1．はじめに

モバイルエージェント技術は、ネットワーク負荷の軽減，非同期分散処理の実現，分散処理系全体への負荷分散の方式として注目されてきた[1]が、サービスモデルとしては情報取得系，あるいはトランザクション性を要求しない更新系が中心であった。しかし、EAI（Enterprise Application Integration）による異業種サービスの統合が進み、また移動体機器の高機能化によるサーバー化が可能になるにつれ、上記のような特徴を持ったモバイルエージェントの活躍の場が広がり、トランザクション型のサービスにも対応することでより幅広いサービス提供が可能になると考える。また、従来はサービス毎にエージェントを作る必要があり、サービス提供者にプログラミング能力が要求されたが、早期提供のためにはサービスロジック作成に注力できる仕組みが必要である。

そのような背景のもと、我々はモバイルエージェントにより異なるサービス群を統合して一つの非同期分散トランザクション型サービスを容易に提供し得るプラットフォームを開発している。本稿ではプラットフォームの機能のうち特にエージェントの移動方式，振舞いの定義方式およびトランザクション制御方式を述べる。

2．プラットフォーム概要

2.1　コンセプト

プラットフォームコンセプトとして、

1. 業種毎に独立に運用されているサービスをインターネットを介して連携制御するシステム基盤を提供する。
2. 連携制御の際には分散トランザクション処理を実現する。
3. 移動体機器に搭載することで、利用者自身がサービス提供者となるような新たなサービス形態を創出する。
4. 統合されたサービス間の連携ロジックはプログラムコードとは分離し、サービスの書換えを容易にする。

ことを挙げる。これにより、後述する旅行プランニングのような代理店型サービスを容易に提供可能になる。

2.2　プラットフォーム構成

プラットフォームはモバイルエージェント（以下、エージェント）部とコア部にわかれ、図2.1のような機能構成となっている。また、図2.2にプラットフォームを用いたシステム構成例と一連の振舞いを示す。本文中の丸囲み数字は図中のシーケンス番号に対応する。

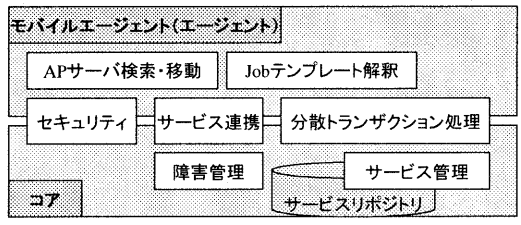

図2.1　プラットフォーム構成

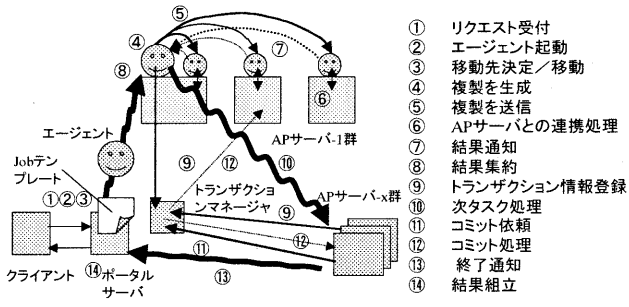

図2.2　プラットフォームによるシステム構成とその振舞い

システムはポータルサーバ，アプリケーションサーバ（以下、APサーバ），トランザクションマネージャ，エージェントおよびクライアントで構成される。コア部はクライアント以外には搭載必須であるが、クライアント側にも搭載することにより、移動体機器からのサービス提供といった新しいサービス形態が可能になる。コア部を搭載したこれらの構成要素は、自らのサービス属性をサービスリポジトリに登録する。サービスリポジトリはお互いに同期が取れており、エージェントが他の構成要素を検出するのに利用される。

ポータルサーバはクライアントからのリクエストを受け付けた後(①)、エージェントを起動し(②)、全処理終了後には結果を組み立てる(⑭)。エージェントは、予めサービス毎にテキストファイル形式で静的に定義されたJobテンプレートと呼ばれるスクリプトを起動時にポータルサーバから渡され、それに従いAPサーバ間を移動し処理をする(③－⑬)。エージェントは、Jobテンプレートを

---

*Proposal of the advanced integrated information service platform using the Mobile Agent.
Kaori FUJINAMI,Toru NAGAOKA,
Technology Development Department , Solution Development Headquarters,
NTT COMMUNICATIONWARE CORPORATION

解釈して処理を行う汎用処理エンジンと言えるので、新規に統合サービスを提供する際にはJobテンプレートを用意すればよく、サービスに特化したエージェントを作る必要はない。

3．エージェントの振舞い

3.1 移動

エージェントはサービスリポジトリより移動先となるAPサーバを検出するが(③)、合致するものが複数あった場合には、そのうちの一つへ移動した後、全てに自分自身の複製を送信し処理を行う(④−⑥)。その後結果を集約してソート及び選択処理をさせることが可能である(⑦,⑧)。これにより、並列処理による処理速度向上のみならず、ポータルサーバへの負荷集中を避け、統合サービスを形成している処理系全体へ負荷分散を図っている。

3.2 Jobテンプレート

Jobテンプレートの構成要素を図3.1に示す。

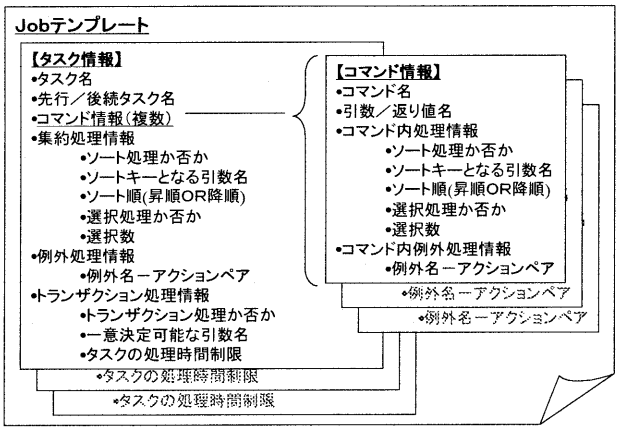

図3.1 Jobテンプレートの構成要素

移動するサービス単位をタスクと呼び、サービスを統合したものをジョブと呼ぶ。タスクは複数のコマンドで構成され、エージェントはコマンドと引数をAPサーバに渡すことにより、サービス固有の処理メソッドにアクセスする。Jobテンプレートにはタスク間の関連を表す先行・後続タスク名，コマンド情報，ソートや選択などの集約処理情報，例外とそれに対応するアクションを関連付けた例外処理情報，トランザクション処理情報が記述されている。コマンドは「flight_reference(旅客機検索)」といった人間が理解できる単位のものから、発話行為理論に基づくエージェント通信言語[2]のような細かい単位での表現も可能であり、汎用性・柔軟性・開発効率といった観点で選択することになる。

引数の名前は、先行ないし同一タスク内のコマンド処理結果を別のコマンドの引数として使用するといったコマンド間での値の引継ぎを考慮して、「ローカル名@コマンド名.タスク名」をグローバル名とし、そこに記されているコマンド名及びタスク名により処理結果から値を探索する。なお、ローカル名はAPサーバ側の内部のコマンド処理で使われるものであり、またコマンドおよびローカル名は同一サービスを提供するAPサーバ間では事前に統一が図れているものとする。

3.3 分散トランザクション処理

トランザクションマネージャ，エージェント，APサーバの協調によって複数タスクの処理に対するAtomicityの保証を実現する。トランザクションマネージャでは各ユーザリクエストに対する各タスクの処理状態や処理結果などをトランザクション情報として管理しており、エージェント自身が処理途中でAPサーバの障害等により消滅しても途中経過は保護することができる。

タスク処理が全て正常終了した場合には、エージェントはトランザクションマネージャに対してコミットを依頼する(⑪)。すると、トランザクションマネージャは管理しているトランザクション情報を用いて、該当APサーバに対してコミット処理を実行する(⑫)。また、ロールバック処理は、エージェントが例外を検知した際にJobテンプレートの例外処理記述に基づきトランザクションマネージャに通知され、コミット時と同様にAPサーバに対してロールバック処理が実行される。

また、タイマー監視機能によりエージェントがデッドロックになることを防ぐ。タイムアウトを検出した時点でJobテンプレート記述の例外処理が行われる。

4．サービスモデル

例えば、旅行プランニングサービスへの適用を考える。航空機(検索・予約)→ホテル(検索・予約)→レストラン(検索)のタスク順で処理が直列的に進むもので、例えばホテルの検索で条件に合うものが存在しない場合には、先行の航空機予約をキャンセルするようなロールバック処理を、またレストラン検索では、先行タスクで決定したホテルの所在地を用いて検索を行うといったサービス間の値の引継ぎを、それぞれ前述の方法により実現している。

5．おわりに

Jobテンプレートによるエージェントの汎用化方式とモバイルエージェントによるトランザクション処理方式を述べた。

参考文献

1) “モバイルエージェントの動向”, 佐藤一郎, 人工知能学会誌 Vol14 No4, pp598-605, 1999
2) FIPA ACL:http://www.fipa.org/