6F－7

自律協調分散システムNoahの通信機構の実現

佐藤文明　小塚　宏　宮崎一哉　福岡久雄

三菱電機（株）情報システム研究所

１．はじめに

分散システムの構築方式として、自律エージェントに基づく方式が注目されるようになってきた。自律エージェントでは、従来のオブジェクト指向におけるオブジェクトがリクエストを一方的に受信するのに対し、自律的な評価機構を持つことが特徴である。自律的な評価機構をもつエージェントに基づくシステムを分散環境に構築する環境として、我々はＮｏａｈ（Network oriented applications harmony）を構築した[1]。

Ｎｏａｈは、大きく通信ブロック、協調処理ブロック、監視制御ブロックに分けられる。ここでは、通信ブロックの設計、及び実現について述べる。

２．ｎｏａｈの通信機構

Ｎｏａｈの目的は、アプリケーションのモジュールであるエージェントが、自律的に動作することを支援することである。そのため、通信機構は自律的に送信／受信することが可能であり、しかも受信においても目的や評価基準に基づいて選択的に受信することができる必要である。

このような目的を実現するために、Ｎｏａｈでは、階層的なタプルスペース通信[2]を採用している。タプルスペース通信は、タプルスペースへのメッセージの投入、取り出しを基本にしており、同期通信、非同期通信、１対１通信、放送通信などの各種の通信機構を提供することができる。また、投入されたメッセージ（タプル）を自律的に検索して取り出すことができるため、エージェントに基づく環境には非常に整合性が良い。

Ｎｏａｈでは、このタプルスペース通信を階層的に分散環境に構築している。その目的は、アプリケーション毎にタプルスペースを利用できること、複数のエージェントによるサービスフィールドの概念を実現するのに適していることによる。

また、Ｎｏａｈでは通信の信頼性を高めるため、複数のノード上にタプルスペースの複製を配置することができる。

Ｎｏａｈの通信機構では、分散環境で動作するエージェント間の通信の他に、エージェントの起動停止をも管理する。すなわち、ＮｏａｈではエージェントはＬａｋｅと呼び、通信機構であるタプルスペースとそのタプルスペースを利用するプロセスを組み合わせて管理する。Ｌａｋｅを作成すると、その中にはプロセスの起動と、通信機構であるタプルスペースの生成が行われる。

３．実現方法

３．１　構成

Ｎｏａｈ通信機構は、分散環境上に存在するタプルスペース管理サーバによって提供される。タプルスペース管理サーバ（ＴＳマネージャ）は、タプルスペース管理、名前管理、マルチキャストＲＰＣ機構の３つの部分から構成される。

（１）マルチキャストＲＰＣ

マルチキャストＲＰＣは、タプルスペース管理サーバとエージェント間の通信に使われる。タプルスペース管理サーバは複数のノード上に存在し、複製を持つタプルスペースを管理している。複製を持つタプルスペースを一貫性を保証しながら更新するには、メッセージの到着順序保存型のマルチキャストＲＰＣ[3]が非常に効果的である。

（２）名前管理

分散環境におけるエージェントの名前を管理するには、管理ポリシに基づいた名前の階層化が有効である。Ｎｏａｈでは、各エージェントはＬａｋｅという単位で名前の管理が行われる。各Ｌａｋｅは、”／”を頂点にした木構造の名前を持ち、”／”によって名前の区切りが与えられる。

また、ここではＬａｋｅに存在するタプルスペ

Communication Mechanism for an Autonomous Cooperative Distributed System : Noah
Fumiaki Sato, Hiroshi Kozuka,
Kazuya Miyazaki, and Hisao Fukuoka
Computer and Information Systems Laboratory
Mitsubishi Electric Corporation
5-1-1 Ofuna, Kamakura 247, Japan

ースの管理ＩＤと、プロセスの動作するノード、プロセスＩＤが管理されている。

（３）タプルスペース管理

タプルスペース管理部は、タプルスペースに存在するタプル、ブロックされているオペレーション、階層の上位／下位のタプルスペースＩＤが管理されている。

３．２　インタフェース

Ｎｏａｈ通信機構では、以下のようなタプルスペースアクセス関数が提供されている。

・Ｎｏａｈ通信のセッション開始
・Ｎｏａｈ通信のセッション終了
・Ｌａｋｅ作成
・Ｌａｋｅ削除
・タプルの投入
・タプルの取り出し（同期／非同期／時間指定）
・タプルの読み込み（同期／非同期／時間指定）
・バルクデータ投入
・バルクデータ取り出し
・バルクデータ読み込み
・ストリームデータの投入
・ストリームデータの取り出し
・ストリームデータの読み込み
・タプルの削除
・Ｌａｋｅの検索
・Ｌａｋｅの状態問い合わせ

この中で、バルクデータに関する処理は、大量のデータを通信する際に、タプルデータのタプルスペースへの投入の完了を待たずに取り出し要求、あるいは読みだし要求に応じるサービスである。

ストリームデータに関するサービスは、タプルスペース通信のようなメッセージベースの通信と、ストリーム型の通信を合成するものである。

４．おわりに

自律協調分散環境Ｎｏａｈの通信機構ついて論じた。Ｎｏａｈでは、アプリケーションのモジュールであるエージェントの自律性を実現するために、階層型のタプルスペースを採用した。また、信頼性の向上、大量データ転送への対応、ストリーム型通信との共存を実現するための設計を行っている。

実際のデータ転送による評価の結果、ＲＰＣと比較して、１対１の通信では２倍程度の通信時間を必要としている。これは、タプルへの集中アクセスがネックになっているためである。これは、分散したタプルスペースでの並行処理により、ある程度改善できると考えられる。また、バルクデータ転送は、ある程度のデータ転送量になると効果的になることが分かった。今後はベースとする通信機構の改良や処理のバッファリングなどで高速化を行う必要がある。

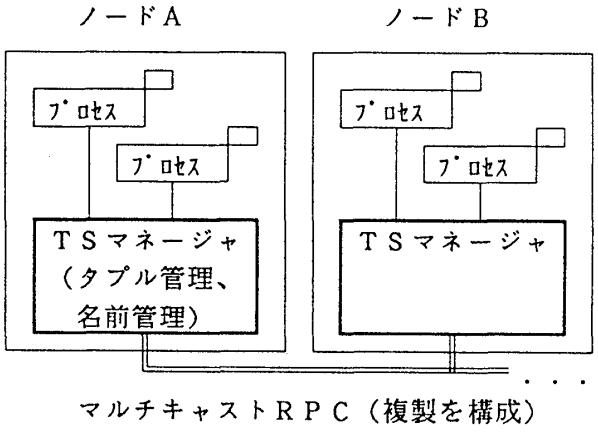

図１　通信機構の構造

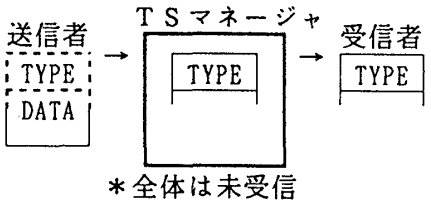

図２　バルクデータの通信

［参考文献］

[1]H.Kozuka, etal.:"An Architecture of Distributed Computing Environment -Noah-,"IEEE 1st International Workshop on Systems Management (1993).

[2]N.Carriero, etal:"Linda in Context," ACM, vol.32, No.4(1989).

[3]K.P.Birman, etal:"Exploiting Virtual Synchrony in Distributed Systems," Proc. of 11th ACM Sysmp. on Operating Systems Principles (1987).