6E－2

Ｊａｖａによる擬似黒板モデルを用いたエージェントの一考察

古川嘉識　山本浩文　藤本憲司
ＮＴＴコミュニケーションウェア

1　はじめに

近年のインターネット技術・WWW技術の普及により、世界中に大量の情報が分散している状況にあり、現在のインターネットの利用者は、この大量の情報を取捨選択しなければならない。そのため、利用者の嗜好に合わせた情報のみを取得したいという要求がおこり、それを解決する１手法として、エージェント技術[1][2][3]が注目を浴びている。

我々はエージェントにとって必要となる基盤技術を確立するため、エージェント間の協調処理を実現する擬似黒板モデルを提案し、エージェントフレームワークを開発した。開発には、高いネットワーク親和性やマルチプラットホームに対応といった特徴を持つ Java を用いた。今回そのフレームワークの有効性をメッセンジャ・システムに適用することで検証する。

2　目的

エージェントとは、

・自律性：各エージェントが価値基準を持ち、それにしたがって行動を決定する。
・社会性：他のエージェントや人間と協調作業する。また人間社会のように同じ目的であっても協調作業する相手は、固定的でなく、動的に変化させることができる。
・反応性：自分の置かれた状況を判断し、その変化に対し柔軟に行動を決定する。
・移動性：ネットワーク上のサーバ間を自律的に移動する。
・知性　：最適処理の選択。ユーザ処理の学習。

といった特徴を備えた、ソフトウェアモジュールである。

我々は、エージェント技術にとって、上記の特徴のうち社会性、移動性に関しては、その基盤となるフレームワークが担う部分であると考えた。そこで、各エージェントが相手を意識することなく協調作業ができ、またエージェントの動的な置き換えが可能なフレームワークを作成した。移動性に関しては、JavaRMI を用いて実現した。

3　擬似黒板モデル

我々は、フレームワーク上のエージェントが協調作業を行うためには黒板モデルが、有効であると考えた。しかし、通常の黒板モデルでは、黒板と呼ばれる共通のデータエリアに各エージェントが非同期的にアクセスすることで協調作業を実現している。そのため、エージェント数が増大した場合、黒板へのポーリング負荷が大きくなり、大規模システムに応用することが難しいという問題が発生する。

そこで、①ポーリングを行わない方法でエージェントの協調作業を行う、②システム動作中にエージェントの追加・削除を容易に可能とする、以下に示す擬似黒板モデルを提案する。

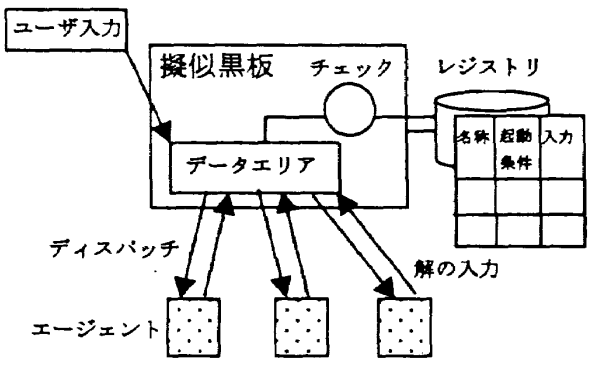

Fig. 1　擬似黒板モデル

擬似黒板モデルでは、通常の黒板モデルのように各エージェントがアクセスして、協調作業を実現するのではなく、黒板からエージェントに黒板の内容をディスパッチし、エージェントはその内容から得た解を黒板に書き込む。これを繰り返すことで、擬似的に黒板モデルを実現する。

起動するエージェントを特定するために、あらかじめ起動可能なエージェントの名称・起動条件・入力情報をレジストリに登録する。

A Study of Agent Using a Pseudo Blackboard Model with Java

Yoshinori Furukawa, Hirofumi Yamamoto, Kenji Fujimoto
NTT Communicationware Corporation

4　フレームワーク

メインサーバには、与えられた作業を遂行するエージェントと処理すべきエージェントを選択し、ディスパッチする擬似黒板が動作している。また、分散サーバには、エージェントが作業を遂行するために必要な各作業モジュールが置かれる。

4－1　エージェント

エージェントは、擬似黒板のレジストリに登録されいてる条件に従って選択され、go メッセージを受けて起動される。起動されたエージェントは、記述されたシナリオにしたがった作業を遂行し、その作業結果を擬似黒板に返却する。このような一連の処理を繰り返すことで、他のエージェントと協調作業を行う。

エージェントは、フレームワークが用意したエージェントメタクラスを継承して作成する。エージェントメタクラスは、レジストリへのエージェント登録・削除のためのメソッドを用意している。エージェントを実行した時に、自動的に擬似黒板への登録が行われる。

また、分散環境への対応は、作業体(作業モジュール)とインタフェース(エージェント)を分離して行う。

4－2　作業モジュール

分散サーバに置かれ、エージェントから起動される。それぞれの分散サーバ上で実際に機能を実現するモジュールである。実行結果をエージェントに返却する。

4－3　メッセンジャシステム

メッセンジャ・システムは、相手の居場所を意識することなく、伝えたい内容を送るものである。このシステムは、相手の居場所の検索と送信の機能から構成されている。居場所の検索方法は、多種多様に考えられるために、随時機能追加・変更できることが望まれる。このような要件持ったメッセンジャー・システムを、今回作成したフレームワークを用いて作成し、期待した動作をするかを検証した。メッセンジャ・システムの概要を、Fig.2 に示す。

相手の場所を探す手段として、ユーザの利用するサーバに last, ping を行う作業モジュールを置く。手順は次の通りである。①擬似黒板に書き込まれた相手名およびメッセージ内容を、検索エージェントへ作業依頼する。②検索エージェントは分散サーバにある作業モジュールを利用し、ユーザを検索しその結果を擬似黒板に書き込む。③次に擬似黒板は送信エージェントに送信作業を依頼する。

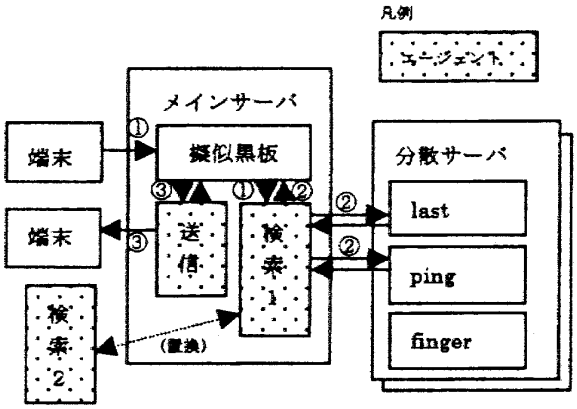

Fig. 2　メッセンジャシステム

5　検証

メッセンジャ・システムのエージェント追加・削除機能を検証するために、分散サーバ上に finger 作業モジュールを置き、last, ping を起動する検索エージェント１から、3 つの作業モジュールを起動する検索エージェント２に置き換えを行った。

このとき、送信エージェントは検索エージェントが削除・追加されたが、検索エージェントを意識すること無く協調作業を続けることができた。

6　まとめ

我々は、擬似黒板モデルを用いたエージェントフレームワークを Java を用いて作成した。そして、そのフレームワーク上にメッセンジャ・システムを構築し、その有効性を検証した。擬似黒板からエージェントをディスパッチすることで、各エージェントが黒板をポーリングすること無く協調作業を実現することができた。また、シナリオエージェントの追加・削除がシステム全体を止めることなく行えることを確認することができた。しかし、解が収束しない場合があるといった黒板モデル自体が持つ問題、シナリオエージェントの追加・削除時の同期の問題は未解決のままであり、今後継続して検討する予定である。

7　参考論文

[1] http://www.trl.ibm.co.jp/aglets/index-j.html
[2] 永井保夫,田原康之,入江豊,大須賀昭彦,本位田伸一：Plangent I インテリジェント・ネットワークエージェント,
http://www2.toshiba.co.jp/plangent/index_j.htm
[3] http://www.firefly.net/