大規模マルチエージェントシミュレーションにおけるプロトコル記述と実行基盤

中島　悠[†a)]　椎名　宏徳[†*]　山根　昇平[†]　八槇　博史[†]
石田　亨[†]

Protocol Description and Platform in Massively Multiagent Simulation

Yuu NAKAJIMA[†a)], Hironori SHIINA[†*], Shohei YAMANE[†], Hirofumi YAMAKI[†], and Toru ISHIDA[†]

あらまし　ユビキタス環境の普及に伴い，都市上の人間を支援する大規模な社会情報システムが実現可能となってきた．このような人間を含む大規模なシステムを検証するために，マルチエージェントシミュレーションを用いるアプローチが考えられる．大規模な社会情報システムをシミュレートするシステムの開発においては，応用領域の専門家と計算機の専門家が協調して作業を行う必要がある．また，大規模な社会情報システムのシミュレーションでは，個々のエージェントが直面する状況も多岐にわたる．更に，都市規模の人間を再現するためには，数十万のエージェントを管理するスケーラビリティが必要となる．本研究では，これらの課題を解決するため，シミュレーションシナリオの処理系とエージェント実装との分離を特徴とする大規模マルチエージェントシミュレーションの実行基盤のアーキテクチャを提案し，その実装・評価を行った．

キーワード　マルチエージェントアーキテクチャ，大規模シミュレーション，シナリオ記述言語

1. まえがき

携帯電話やカーナビゲーション等の移動通信端末の普及，GPS を代表とする測位システムの精度向上などに伴って，都市上の人間を支援する大規模な社会情報システムが実現可能となってきた [1]．その例として，個々の通行者が携帯する端末を通じて経路指示を行う大規模なナビゲーションシステムが考えられる [2]．そこでは，群衆に対して同じ情報をブロードキャストする誘導ではなく，それぞれの通行者に対する個別の誘導が可能となる．

このような人間を含む社会情報システムの挙動を調べるには，人間を用いた実証実験を行うことが望ましい．しかし，都市という広大な空間で大量の被験者を用いた実験を行うことは非常に困難であり，コストも大きい．そこで，人間をモデル化したエージェントを用いるマルチエージェントシミュレーションにより，システムを分析するアプローチが考えられる．これまでに，エージェントに与えるプロトコル(注1)を設計してシミュレーションを行う手法が提案され，避難誘導を題材としてその有効性が検証されている [3]．本研究では，人間の行動を反映したエージェントシミュレーションを都市規模（数十万）の人間を支援する社会情報システムの分析に適用することを目標として，大規模マルチエージェントシミュレーション環境を実現することを目指す．本論文では，数十万のエージェントをプロトコル記述により制御できるアーキテクチャを提案し，その実装・評価について述べる．

我々が取り組んだ課題は以下の 3 点である．

（1）　プロトコル設計とエージェント開発の分離

社会情報システムのシミュレーションにおいては，応用領域の専門家が，システムの利用者をモデル化したエージェントに与えるプロトコルを設計する必要がある．しかし，応用領域の専門家は計算機の非専門家

† 京都大学情報学研究科社会情報学専攻，京都市
Department of Social Informatics, Kyoto University, Kyoto-shi, 606–8501 Japan
* 現在，富士通株式会社
a) E-mail: nkjm@ai.soc.i.kyoto-u.ac.jp

(注1)：本論文では，プロトコルとは，エージェントと外界（他のエージェントや環境）とのインタラクションを記述したものを指す．

であることが多く，高度なプログラミング技術を要するエージェントシステムの開発は困難である．そこで，応用領域の専門家がプロトコルを設計し，計算機の専門家がエージェントシステムの開発を行う環境を作成することで，異なる領域の専門家が協調作業できる体制を用意する．

（2） プロトコルの切換

都市上の社会情報システムのシミュレーションでは，個々の利用者エージェントが直面する状況も多岐にわたる．例えば，都市上を移動するエージェントを作成するときに，エージェントが街中で直面するあらゆる状況におけるプロトコルを記述したならば，それは長大で複雑なものとなってしまう．そこで，プロトコル設計者が注目したい特定の状況におけるプロトコルを複数用意し，それを状況に応じて切り換えることで，一つひとつのプロトコルを簡潔にするアプローチをとる．そのため，システム動作時でもエージェントに与えるプロトコルを変更できるようなアーキテクチャとする必要がある．

（3） スケーラビリティの確保

従来のプロトコル処理系やエージェントシステムは大規模なエージェントの管理を想定していなかった．大規模な社会情報システムのシミュレーションを行うには，人間をモデル化したエージェントをシミュレータ上で大量に制御する必要がある．我々は，近年開発されたイベント駆動オブジェクトを用いた大規模エージェントサーバを応用することでスケーラビリティの確保を実現する．

以下，**2.** では，上記の三つの課題を解決する大規模マルチエージェントシミュレーション基盤のためのアーキテクチャを提案する．**3.**，**4.** では，基盤技術と実装したプラットフォームについて述べる．**5.**，**6.** では，プラットフォームの応用例と評価を示す．

2. アーキテクチャ

エージェントに与えるプロトコルを設計してエージェント内部モデルを制御する方式は，プロトコル記述とエージェント内部モデルを統合したものをエージェントとして実装する方式（図 1）と，外部のプロトコル処理系がエージェント内部モデルを制御する方式（図 2）の二つに大別される．

図 1 の方式では，エージェントシステム開発者が，AgentUML [4] のような実行可能でない言語で記述されたプロトコル記述をエージェント内部モデルと統合

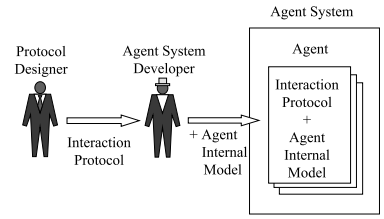


図 1 エージェント内にプロトコルと内部モデルを実装
Fig. 1 Both protocol and internal model are implemented in each agent.

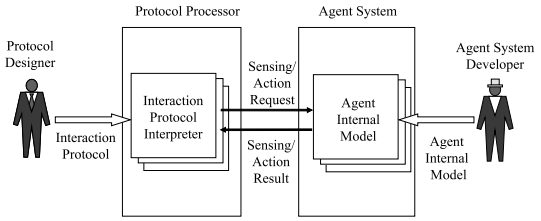


図 2 外部のプロトコル処理系がエージェントシステムを制御
Fig. 2 External protocol interpreter controls agent system.

し，それをエージェントに実装する．プロトコル記述とエージェント内部モデルがともに一つのエージェントに実装されるこの方式では，各領域の専門家の知見をいったんエージェントシステムの開発者が吸収し，それをエージェントの実装に反映するという手順を踏む必要があるため，開発やシミュレーション実施における効率の面で問題が生じる．また，システム動作時に，状況に応じてエージェントに与えるプロトコルを切り換えることも困難である．

図 2 の方式では，実行可能なプロトコル記述言語によってプロトコルを記述し，それを外部の独立した処理系であるプロトコルインタプリタによって解釈・実行させることでエージェントを制御する．この方式では，エージェントの実装や内部モデルを意識することなくプロトコルを設計できるため，プロトコル設計とエージェント開発が分離可能であり，複数の専門家が協調してマルチエージェントシステムを開発することができる．

本研究では，図 2 のアーキテクチャにスケーラビリティをもたせる拡張として，プロトコルインタプリタとエージェントの内部モデルの両方を大規模エージェントサーバ上に実現することを提案する（図 3）．大規模エージェントサーバは，エージェントをオブジェクトとして保持し，動作させるオブジェクトに対し

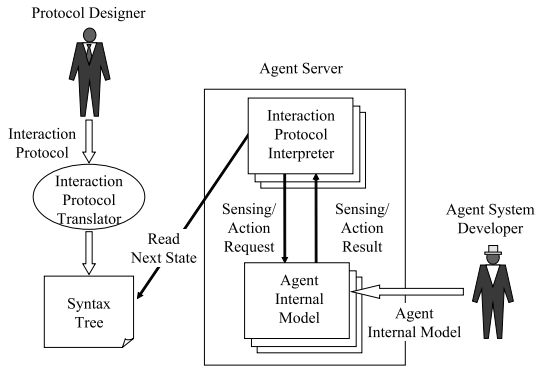


図 3　エージェントシステム上のプロトコルインタプリタがエージェントを制御
Fig. 3　Protocol Interpreter on agent system controls agent.

て適宜スレッドの割当を行う方式をとることで，数十万のエージェントを管理することを可能としている．大規模エージェントサーバの例として，後に述べる Caribbean [5] が挙げられる．

この方式では，図 2 のアーキテクチャと同様にプロトコル記述とエージェント開発が分離されているため，プロトコル設計者はエージェント実装の細部にとらわれずにプロトコルを設計することができる．また，システム動作時にシナリオの解釈・実行を行うインタプリタが，エージェント内部モデルに対してセンシング/アクションの実行を依頼し，その結果を受けとる構成となっている．そのため，システム動作時でもエージェントに与えるプロトコルを切り換えることができる．つまり，本アーキテクチャでは図 2 のアーキテクチャの利点を維持できているといえる．

他の大規模なマルチエージェントプラットフォームとして，MACE3J [6]，MadKit [7]，Robocup Rescue [8] が挙げられる．

MACE3J は，汎用的なエージェントプラットフォームであり，分散環境に対応することでスケーラビリティを上げることを主目的としている．MadKit は，特有の社会モデル上にエージェントシステムを設計するプラットフォームであり，エージェントのアーキテクチャやエージェント間の通信手段の差異を吸収することを目的としたものである．Robocup Rescue は，大規模災害救助シミュレーション用のプラットフォームであり，それに特化することで最良のパフォーマンスを得ることを目的としている．

しかし，これらのプラットフォームでは，エージェントとそれに与えるプロトコルが明示的に分離されていない．それに対して，我々のアーキテクチャは，エージェントとプロトコルを明示的に分けることで，計算機の専門家と応用領域の専門家が協調しながら，プロトコルに注目したシミュレーションを開発する環境を提供することが目的である．

3. 基盤技術

本章では，シミュレーションプラットフォーム構築に用いた基盤技術，シナリオ記述言語 Q [9] と大規模エージェントサーバ Caribbean について説明する．

3.1 シナリオ記述言語 Q

Q は，エージェントの内部メカニズムには言及せず，エージェントと外界とのインタラクションプロトコルを，シナリオとして記述するための言語である．人間行動のモデル化において，人間行動の中身を記述する旧来のエージェント記述方式ではなく，Q のように人間と外部世界とのインタラクションプロトコルの様態をそのままシナリオとして記述する方式が効果的であることが示されている [10]．Q の特徴的な言語機能として，以下の 3 点が挙げられている．

- センシングとアクション

センシングは，インタラクションのきっかけとなるイベントの観測をエージェントに依頼するために用いる．そのため外界への副作用はなく，センシングは，きっかけとなるイベントが観測されるまで続けられる．一方のアクションは，外界に作用する振舞いをエージェントに依頼するために用いられる．なお，センシングとアクションの記述はそれぞれ？と！の記号から始まる．

- ガード付きコマンドとシナリオ

ガード付きコマンドは，複数のイベントを並行に観測するために導入されている．観測中のイベントのうち一つでも観測されれば，後続するアクションを実行する．シナリオは状態遷移表現を用いて記述され，各状態はこのガード付きコマンドとして定義される．

- エージェント

エージェントには何をすべきかが記述された行動シナリオが与えられ，エージェントはそれに基づいて行動する．

また，Q にはシナリオ記述のサポートのために，インタラクションパターンカード（IPC）と呼ばれるツールが導入されており，計算機の非専門家でも容易にシナリオの記述ができるようになっている．

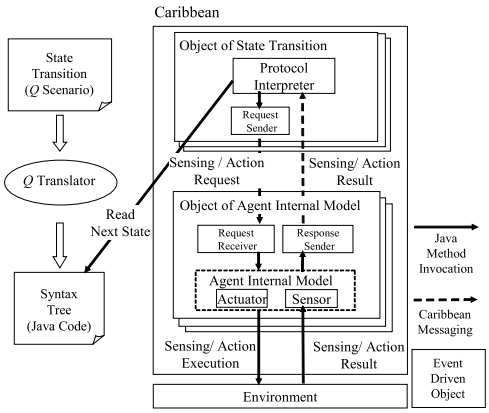


図 4　Caribbean/Q の概観
Fig. 4　Overview of Caribbean/Q.

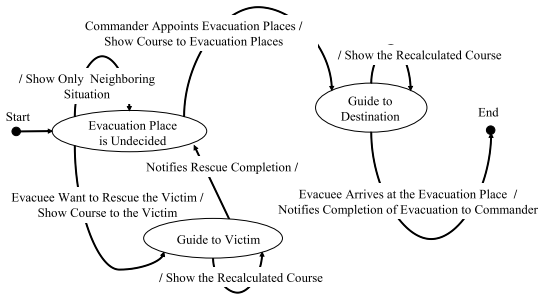


図 5　誘導者エージェントのプロトコル例
Fig. 5　Example of a guide agent protocol.

3.2 Caribbean

Caribbean は Java 言語で実装された大規模なエージェントサーバである．Caribbean はエージェントをオブジェクトとして管理する．Caribbean 上には，サービスオブジェクトとイベント駆動オブジェクトの 2 種類が存在する．Caribbean 上のオブジェクト間は，Caribbean メッセージングにより通信を行う．サービスオブジェクトとは，常に動作可能なオブジェクトで，共通の情報をもつデータベースのように頻繁にアクセスされるモジュールを実装するためのオブジェクトである．それに対して，イベント駆動オブジェクトは，他のオブジェクトからメッセージを受け取るときのみ Caribbean スケジューラからスレッドを割り当てられ，動作するものである．システム上の多くのモジュールは，このオブジェクト上に実装される．

全オブジェクトにスレッドを割り当て，それらを並行的に動作させる方式では，数百程度のオブジェクトしか動作させることができない．Caribbean ではすべてのオブジェクトを並行的に動作させるのではなく，動作させたいイベント駆動オブジェクトに対して適宜スレッドの割当を行う方式をとることで，大量のオブジェクトを動作させることを可能としている．

また，Caribbean では，オブジェクトをメモリと補助記憶との間で出し入れすることで，メモリに呼び出すオブジェクト数を制限し，メモリの消費量を抑えている．アプリケーションを実行しているときに，メモリに呼び出すオブジェクトの数が一定数を超えた場合，Caribbean はメッセージ処理を行っていないオブジェクトを補助記憶へ移す．この補助記憶に移されたオブジェクトに対して他のオブジェクトからメッセージ送信が行われた場合，そのオブジェクトはメモリ上へ再び呼び出され，メッセージの処理を行う．このスワッピングを効率的に行うことで，Caribbean はメモリ上に保持できない数のエージェントをも管理できる．

4. 実　　装

4.1 Caribbean/Q の構成

我々は，**2.** で述べたアーキテクチャに基づき，シナリオ記述言語 Q と大規模エージェントサーバ Caribbean を用いて，大規模マルチエージェントシミュレーションプラットフォームを実装した．そのプラットフォーム（以下，Caribbean/Q）の概観を図 4 に示す．

Q シナリオには，エージェントと外界とのインタラクションプロトコルが記述される．例えば，災害時に避難者を誘導する人をモデル化したエージェントのプロトコルとして，図 5 に示される状態遷移記述を考える．この状態遷移記述の一部を Q 言語を用いて記述すると，図 6 の破線で囲まれるような Q シナリオとなる．この Q シナリオを Q トランスレータに入力し，Caribbean 上の状態遷移機械オブジェクトが読み取れる形式の構文木へと変換する．状態遷移機械オブジェクトは変換された構文木を逐次実行することで，Q シナリオで記述されたプロトコルを実行する．このように，Q の処理系を Caribbean のイベント駆動オブジェクトとして取り込むことで，Caribbean のスケーラビリティを活用した．

4.2 シナリオの実行

プロトコルインタプリタとエージェント内部モデルはともに，Caribbean 上のイベント駆動オブジェクトに実装される．一つのエージェント内部モデルオブジェクトに対して，一つの状態遷移機械オブジェクト

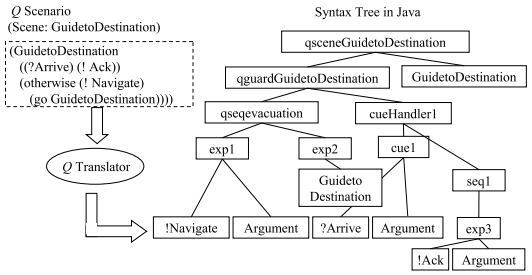


図 6　Q シナリオを Q トランスレータにより構文木に変換
Fig. 6　Q Scenario is translated to the corresponding syntax tree using Q translator.

が生成される．もう一つ別の状態遷移機械オブジェクトを生成して，それをエージェントに割り当てることで，エージェントに与えるプロトコルを動的に切り換えることができる．

シナリオの実行要求が状態遷移機械オブジェクトに対して行われると，状態遷移機械オブジェクトとエージェント内部モデルオブジェクトの間でメッセージの交換が始まる．最初に，状態遷移機械オブジェクトは，センシング/アクションの実行依頼をエージェント内部モデルオブジェクトに対して Caribbean メッセージとして送る．続いて，エージェント内部モデルオブジェクトは，センシング/アクションの実行を環境に対して行い，その実行結果を Caribbean メッセージとして状態遷移機械オブジェクトへと通知する．最後に，通知を受けた状態遷移機械オブジェクトは Q トランスレータにより変換された構文木を読み込んで，次状態へと遷移する．これらの処理を繰り返すことで，シナリオを実行する．

ここで，状態遷移機械オブジェクトはシナリオ実行の際に，エージェント内部モデルオブジェクトに対して，センシング/アクションの実行依頼メッセージを送り，実行完了メッセージを受け取ることを繰り返すのみである．つまり，このメッセージングを処理するという条件さえ満たしていれば，エージェント開発者は自由にエージェントを開発することができる．

また，エージェント内部モデルオブジェクトと状態遷移機械オブジェクトが分離しているため，シミュレーション実行時にエージェントに与えるプロトコルを切り換えることができる．これにより，エージェントにその場に応じたシナリオを与えることが可能となる．

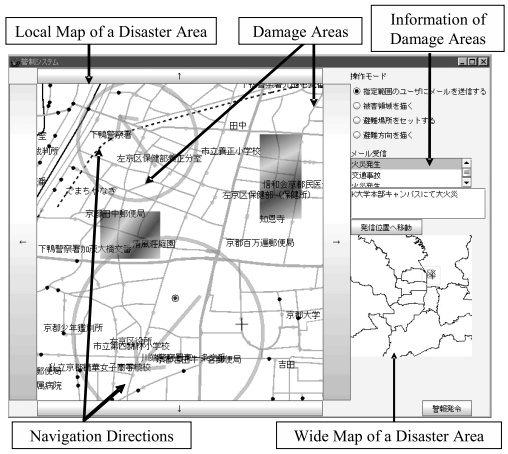


図 7　大規模避難誘導システムスクリーンショット
Fig. 7　Screenshot of large-scale evacuation navigation system.

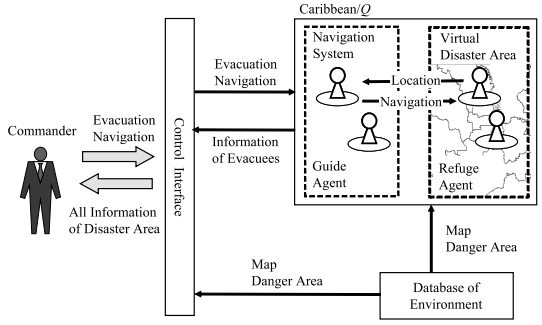


図 8　大規模避難誘導システムのシミュレーション
Fig. 8　Simulation of large-scale evacuation navigation system.

5. 応 用 例

本章では，Caribbean/Q を用いた社会情報システムのシミュレーション例について述べる．我々は，簡単な応用例として，仮想的な社会情報システムである大規模避難誘導システムと，そのシステムの仮想ユーザである避難者エージェントを生成するシミュレータを作成した．

試作した誘導システムは，管制官が図 7 のような管制インタフェースを通じて，GPS 付携帯電話をもっている避難者を誘導するものである．管制官は地図上の誘導エージェントに対して大局的な指示を行い，誘導エージェントが避難者に対して個別の誘導を行う．システムの構成を図 8 に示し，主要なモジュールについて説明する．

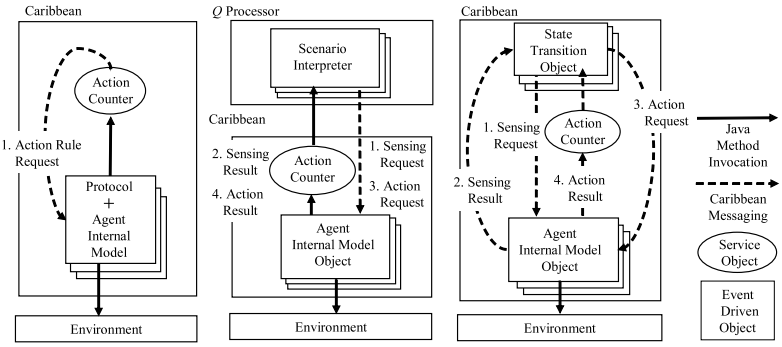


図 9　評価に用いたシステムの構成
Fig. 9　Configuration of the system for evaluation.

● 管制インタフェース　管制官は，管制インタフェースを通じて誘導エージェントに対して避難方向を指示する．管制インタフェースには広域の地図が表示され，避難者の位置情報を俯瞰できる．更に，管制官はインタフェースを通じて，避難場所や要救助者位置を地図上に指定することや，火災などの危険地域に関する情報を地図上に記録することができる．

● 誘導エージェント　誘導エージェントは，仮想災害地域上に存在する避難者エージェントに対して誘導を行う．誘導エージェントは，管制官から指示された誘導方向に存在する最寄りの避難場所を避難者エージェントに提示する．また，近隣に要救助者がいるときには，その場所を提示する．

● 避難者エージェント　避難者エージェントは，誘導システムの仮想的なユーザである．避難者エージェントは，下記のシナリオに基づいて避難行動をとる．

```
(defscenario disasterhappen ()
   (evacuation      ;; 避難状態
     ((?destination) (!navigate)))
;;  避難場所が指定されたら，そこへ向かい，シナリオを終了.
      ((?victime) (!rescue) (go evacuation))
;;  要救助者の位置が指定されたら，そこへ向かい，避難状態へ.
      (otherwise (!wander) (go evacuation))))
;;  それ以外のとき，自分の行きたい方向へ移動し，避難状態へ.
```

この応用例では，避難者エージェント群に単純な単一のシナリオを与えた．消防士や警察といった個人の社会的な役割を反映したシナリオや，怪我の有無といった個人のおかれている状況を反映したシナリオを用意することで，より多様な人間が存在する状況でのシミュレーションも可能である．

このシミュレーションにおいて，1 万体のエージェントが 1 アクション動作するために要した時間はおよそ 1 秒であった．

6. 評　　価

本章では，Caribbean/Q の性能評価を行う．本評価実験では Xeon3.06 GHz のデュアル CPU と 4 GB のメモリを備えたマシンを用いた．

6.1 Caribbean/Q のスケーラビリティ

第 1 の評価として，Caribbean/Q のスケーラビリティを計測する．そのために，Caribbean 単体の場合（図 9 (a)），外付けしてエージェントを制御する場合（図 9 (b)），Caribbean/Q の場合（図 9 (c)）において，エージェント数の変化が処理能力に与える影響を調べる．

Caribbean は完全にイベント駆動で動作するため，同一のイベント駆動オブジェクトが繰り返し動作することが起こり得る．そこで，各エージェントの動作回数を均等にするために，アクションカウンタにより全イベント駆動オブジェクトが動作したことを確認してから，次状態の処理を行わせるシステム構成とした．

エージェントには，下記のシナリオ(注2)と単純なセンシング/アクションを与えて，評価実験を行った．

(注2)：複雑なシナリオでは，状態数が増加し，並行処理するセンシングが増加する．しかし，状態遷移機械オブジェクトにおいて状態の移動は構文木の枝を一段たどるだけであるため，シナリオ中の状態数が増加しても処理速度に影響を与えない．また，並行処理するセンシングが増加することは，エージェント内部モデルオブジェクトから返されるセンシングの名前を表す変数のパターンが増えるだけであり，処理速度に影響を与えない．

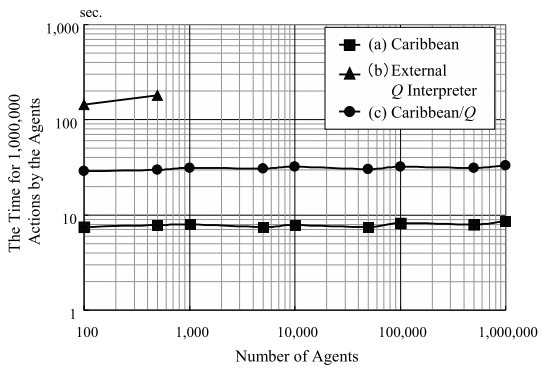


図 10　プラットフォーム評価結果
Fig. 10　Evaluation result of scalability of platform.

```
(defscenario scenario ()
  (scene1
    ((?receive) (!send) (go scene1))))
```

図 9 の各方式において，エージェント数とエージェント群が 100 万アクションを実行したときの処理時間との関係をグラフにしたものを，図 10 に示す．

Q インタプリタを Caribbean に外付けする結合方式では，処理速度が遅く，管理できるエージェント数も数百体程度であった．これは，プロトコル処理系とエージェントシステムとの間でプロセス間通信が発生すること，今回用いたプロトコル処理系の並行処理が OS のマルチスレッド機能に依存していたことが原因である．

それに対して，Caribbean/Q では，100 万体のエージェントを管理することができた．更に，Caribbean/Q では，Caribbean のスケーラビリティが維持され，管理するエージェント数の増加がアクションの処理時間に大きな影響を与えていないことが分かる．これは，Caribbean 上に実装されたシナリオ処理系に Q シナリオを変換した Java コードを読み込ませることで，シナリオ処理とエージェント処理の両方を Caribbean のスレッド管理の対象としたためである．

6.2　Caribbean と Caribbean/Q の比較

第 2 の評価として，Caribbean と Caribbean/Q の処理時間を比較することで，システム開発の容易さと処理速度のトレードオフについて検討する．

Caribbean では，高速処理やスケーラビリティを実現するために，エージェントの構成に必要なデータや機能を一つのイベント駆動オブジェクト上に実装することが求められている．それに対して，本プラットフォームでは，プロトコル記述とエージェント内部モデルの分離及びプロトコルの切換のために，それが状態遷移機械オブジェクトとエージェント内部モデルオブジェクトの二つに分離されている．そのため，マルチエージェントシミュレーション開発における二つの利便性と，処理メッセージ数の増加がトレードオフとなる．Caribbean 単体では，エージェントが 1 回のアクションをするために 1 回のメッセージングを行う必要があるのに対して，Caribbean/Q では，エージェントが 1 回のアクションをするために 4 回のメッセージング（センシングの実行依頼・結果通知，アクションの実行依頼・結果通知）を行う必要がある．

つまり，Caribbean と Caribbean/Q が同一数のアクションを処理するために要する時間の比 P は，下記の式で表される．

$$P = \frac{4m + s + a}{m + s + a} = \frac{4 + \frac{s+a}{m}}{1 + \frac{s+a}{m}} \qquad (1)$$

ここで，m は Caribbean メッセージングの平均処理時間，s はセンシングの平均処理時間，a はアクションの平均処理時間である．

5. で述べた応用例においては，$a + c \fallingdotseq 4m$ であったので，$P \fallingdotseq 1.6$ となる．

7. むすび

本研究では，大規模マルチエージェントシミュレーション基盤のためのアーキテクチャを提案した．更に，そのアーキテクチャに基づいて Caribbean/Q を実装・評価し，応用例を示した．

本研究において，我々が取り組んだ課題は以下の三つである．

（1） プロトコル設計とエージェント開発の分離

本アーキテクチャを用いることで，プロトコル設計とエージェント開発を分離することができる．これにより，複数の専門家が協調して大規模マルチエージェントシミュレーションを実施できる環境を実現した．

（2） プロトコルの切換

プロトコル処理系とエージェント内部モデルを分離した構成とすることで，シミュレーション実行時でもプロトコルを切り換えることができる．これにより，エージェントに対して状況に応じたシナリオを割り当てることを可能とした．

（3） スケーラビリティの確保

大規模エージェントサーバ上にプロトコル処理系と

エージェント内部モデルを取り込むことで，プロトコル記述によりエージェントを制御するシミュレーション環境のスケーラビリティの問題を解決した．

評価結果から，本研究で実装した Caribbean/Q により，目標とする数十万エージェントを用いたシミュレーションが実施可能であることが確認された．しかし，その一方で，シミュレーションの効率的運用のためには，処理の更なる高速化が必要である．また，大規模シミュレーションの実行過程の可視化や，マルチエージェントシミュレーションを用いた大規模な社会情報システムの分析手法についても検討していく予定である．

謝辞 日本 IBM 東京基礎研究所山本学氏，田井秀樹氏，数理システム山本晃成氏に様々な御協力を頂いたことを感謝します．本研究は，日本学術振興会科学研究費基盤研究（A）(18200009，2006-2008)，戦略的情報通信研究開発推進制度地域情報通信技術振興型研究開発（2005–2007）の助成を受けて行われた．

文　献

[1] 幸島明男，和泉憲明，車谷浩一，中島秀之，“ユビキタス計算環境におけるコンテンツ流通のためのマルチエージェントアーキテクチャ：consorts，” 人工知能誌，vol.19, no.4, pp.322–333, 2004.

[2] T. Ishida, “Society-centered design for socially embedded multiagent systems,” Cooperative Information Agents VIII, 8th International Workshop (CIA-04), pp.16–29, 2004.

[3] Y. Murakami, T. Ishida, T. Kawasoe, and R. Hishiyama, “Scenario description for multi-agent simulation,” Proc. second international joint conference on Autonomous agents and multiagent systems (AAMAS-03), pp.369–376, 2003.

[4] J. Odell, H.V.D. Parunak, and B. Bauer, “Representing agent interaction protocols in UML,” AOSE, pp.121–140, 2000.

[5] G. Yamamoto and H. Tai, “Performance evaluation of an agent server capable of hosting large numbers of agents,” AGENTS-01, pp.363–369, ACM Press, New York, NY, USA, 2001.

[6] L. Gasser and K. Kakugawa, “Mace3j: Fast flexible distributed simulation of large, large-grain multi-agent systems,” The First International Joint Conference on Autonomous Agents & Multiagent Systems (AAMAS-02), pp.745–752, Bologna, ACM, 2002.

[7] O. Gutknecht and J. Ferber, “The madkit agent platform architecture,” Agents Workshop on Infrastructure for Multi-Agent Systems, pp.48–55, 2000.

[8] T. Takahashi, I. Takeuchi, T. Koto, S. Tadokoro, and I. Noda, “Robocup rescue disaster simulator architecture,” Workshop Working Notes on RoboCup Rescue (ICMAS-2000), ed. H. Kitano, S. Tadokoro, K. Fischer, and A. Burt, pp.97–105, 2000.

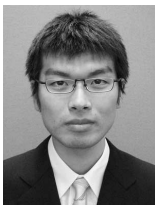
[9] T. Ishida, “Q: A scenario description language for interactive agents,” Computer, vol.35, no.11, pp.42–47, 2002.

[10] Y. Murakami, Y. Sugimoto, and T. Ishida, “Modeling human behavior for virtual training systems,” Proc. Twentieth National Conference on Artificial Intelligence (AAAI-05), pp.127–132, 2005.

（平成 18 年 1 月 18 日受付，4 月 26 日再受付）

中島　悠

平 18 京都大学大学院情報学研究科社会情報学専攻修士課程了．現在，同大学院博士課程在学中．学振特別研究員．大規模マルチエージェントシステム，マルチエージェントシミュレーションに関心をもつ．



椎名　宏徳

平 18 京都大学大学院社会情報学専攻修士課程了．同年，富士通（株）入社．大規模ナビゲーションシステムに興味をもつ．

山根　昇平

平 17 京大・工・情報学科卒，現在，同大大学院社会情報学専攻修士課程に在学中．マルチエージェントシステムに関心をもつ．

八槇　博史　（正員）

京都大学大学院情報学研究科社会情報学専攻博士後期課程了．博士（情報学）．京都大学情報学研究科助手を経て，現在，同研究科講師．マルチエージェントシステム及びそのネットワーク応用に興味をもつ．

石田　亨　（正員）

昭 51 京大・工・情報工学卒，昭 53 同大大学院修士課程了．同年日本電信電話公社電気通信研究所入所．現在，京都大学大学院情報学研究科教授．工博．人工知能，社会情報システムに興味をもつ．