マルチエージェントモデルのグラフ表現に基づく シミュレーションプログラム開発環境

A Programming Environment for Multi Agent Simulation Based on Graph Representation

木村 慶史 (大阪大学大学院 情報科学研究科)

内容梗概

本研究ではマルチエージェントシミュレーションのプログラムソースの開発を支援するプログラム開発環境を提案する。近年、様々なマルチエージェントシミュレーションツールが提案されている。しかし、シミュレーションモデルの表記法が定義されておらず、また、全てのツールは独自のプログラミング言語を採用しているため、プログラミングスキルの低い分析者にとってはプログラムソース作成に多大な労力や時間を要している。

そこで、シミュレーションモデルの記述法をグラフ表現に基づいて定義し、支援インタフェースを用いて分析者が容易にプログラム開発が行えるようにする。さらに、分析者が使用するマルチエージェントシミュレーションツールに対応したプログラムソースを自動生成することで分析者への負担を軽減させる。シミュレーションモデルの構造情報のみでは決定できない演算の順序などのプログラミング上の設定情報についても、処理順序設定支援手法により処理順序の設定を自動化することで、この負荷を軽減する。

提案するプログラム開発環境の有用性を確認するため、被験者 2 人によるプログラムソースの作成・修正実験を行った。実験結果から提案するプログラム開発環境を使用するとプログラムソースの作成・変更にかかる時間をテキストエディタを用いる場合と比べて平均69%短縮することができることを示す。

キーワード

マルチエージェントシミュレーション, グラフ表現, プログラム開発環境

1 序論

近年、生物、生態系、社会、経済といった複雑系 [1][2] に対するシミュレーション手法としてマルチエージェントシミュレーション (multi-agent simulation:以下、MAS)[3][4] が注目されている。MAS とは、人間のような認識・判断を行い、自律的に行動する主体をエージェントし、複数のエージェントが相互作用することで現れる振る舞いを分析するための手法である。エージェントはある基準に従って行動し、他のエージェントに影響を与える。このような単純な局所的ルールが社会といった系全体を構成する。MAS の研究が進行し、様々な MAS ツール (Swarm[5], Repast[6], StarLogo[7], artisoc[8] など) が開発され、これらを用いた多数のシミュレーションが行われている [9][10][11]。しかし、現状では MAS の特徴に対応したモデル記述法が提案されておらず、分析者は問題をどのようにシミュレーションモデルとして記述すればいいかということに時間を割かなければならない。さらに、プログラムソースの記述においては分析者が使用する MAS ツール固有のプログラミング言語を用いて記述する必要がある。分析者にとって使用するツールに応じてプログラミングスキルを修得するためには多大な労力や時間を要する。

そこで、本研究ではモデル記述およびソースコード作成を容易にし、MAS のプログラム開発を支援するプログラム開発環境を提案する。本開発環境は分析者が行うシミュレーションモデル記述および、モデルを基にしたプログラムソース生成を支援する。分析者は本開発環境のモデルエディタ上で、本研究で提案する有向グラフ表現を用いてモデルを記述する。また、プログラム上では、エージェント間の相互作用の処理順序によって結果が異なることから、分析者は意図する処理順序の設定を行う必要がある。しかし、この設定作業は分析者に負荷のかかる作業であるため、グラフ構造を用いた設定支援方法を用いて設定項目数の削減を図る。分析者によって記述されたモデルおよび処理順序設定を用いてプログラムソースを自動出力し、出力されたプログラムソースを MAS ツール上で実行することでシミュレーションを行うことができる。

2 マルチエージェントシミュレーションプログラム開発環境の構成

2.1 開発環境の概要

MAS の開発は一般的に図 2.1 に示すフローで行われる。シミュレーションを行う分析者は、シミュレーション対象の動きをおおまかなに持っており、これを具体化する形で MAS モデルとして表現する。表現されたシミュレーションモデルに基づいて、プログラムソースを記述し、そのプログラムソースをプログラム実行環境である MAS ツール上で動作させる。

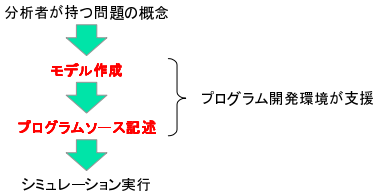


図 2.1: MAS 実施フロー

本フローにおいてシミュレーション実行は前述の MAS ツールを用いることができるが、現状ではシミュレーションモデルの表現、プログラムソースの記述を支援するツールなどは提供されていない。よって、本研究で提案するシミュレーションプログラム開発環境はこれらの作業を支援する。

図 2.2 に、以上の開発フローに基づいたシミュレーションプログラム開発環境の構成を示す。まず、分析者は MAS モデルエディタを用いて分析対象となる問題のシミュレーションモデルを作成する。作成されたシミュレーションモデルは XML(Extensible Markup Language) 記述を用いてモデルに内在する様々な情報が保持される。プログラムソースジェネレータは作成された XML ファイルを解釈し、シミュレーションプログラムソースを生成する。本研究では、MAS 実行環境として、日本語を用いたプログラム記述に対応した artisoc を使用する。生成されたプログラムソースは MAS ツールを用いてシミュレーションが実行され、シミュレーション結果が出力される。

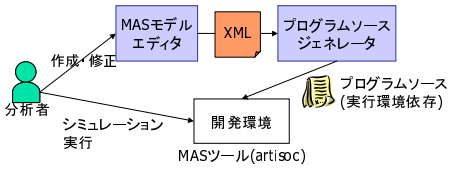


図 2.2: シミュレーションプログラム開発環境の構成

ここで、提案するプログラム開発環境がない場合でも、分析者は一般的に問題対象を整理するために、ノードやアークを用いたグラフ構造のモデルを作成する。また、エージェント間の相互作用の発生順序によって結果が異なってくるため、プログラミングソースを作成する際に、処理順序を考慮する。これを踏まえ、本研究では、プログラム開発環境上で作成するモデルを有向グラフを用いたモデル構造とし、モデルの構成要素を定義する。また、処理順序を決定するのは分析者にとって負荷のかかる作業であるため、モデルの構造を用いた処理順序の設定支援を行う。モデルの構造情報および順序設定情報を含んだ XML データをプログラムソースジェネレータに入力することで、プログラムソースを自動生成する。

2.2 MAS モデルエディタ

マルチエージェントモデルでは、エージェントとその相互作用を表現する必要がある。エージェント間の相互作用はイベントを用いて表現する。図 2.3 に示すように、エージェントおよびイベントは状態や性質を示す “属性” を持つ。エージェントが持つ属性は条件によりイベントを発生させ、そのイベントが他のエージェントへ “影響” を与える。影響を受けたエージェントは自身の属性を変更する “演算” を有し、その結果さらに他のエージェントに “影響” を与える。

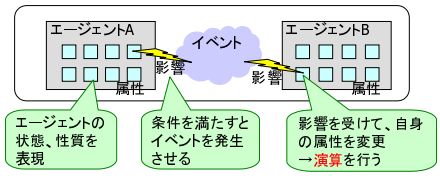


図 2.3: 実世界の事象の概念図

以上をまとめると、MAS モデルエディタ上で記述すべきものは “属性”、“影響”、“演算” の 3 点であり、エージェントおよびイベントの記述は、それぞれを “変数ノード”、“アーク”、“演算ノード” の 3 要素とし、これらを用いた有向グラフでモデルを記述する。

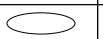
2.2.1 シミュレーションモデル構成要素

本研究で提案するマルチエージェントモデルの 3 つの構成要素について説明する。

- 変数ノード

  値を保持するためのノードであり、エージェントやイベントの状態や性質を表現するために使用する。影響を受けて値を変化させるものや、不変であるもの、論理値を取るものの 3 種類を定義する。変数ノードは、楕円内に変数名を表記し、3 種類の変数ノードの表記法を表 3.1 に示す。

表 3.1: 変数ノード表記法

変数ノード表記	種類
	可変
	不変
	論理値

- 演算ノード

  アークによって伝播された値を用いて演算を行うためのノードである。演算には条件式を用いた四則演算を行うことができる。図 2.4 の演算ノードの使用例にあるように、演算ノードは正円で表記する。図中に次に説明するアークが描かれているが、ここでは値を伝播する要素と解釈してよい。本例の場合、演算ノードに x および y という値が伝播され、x の値によって演算式が決定され、それぞれの場合で伝播する演算結果を変更していることが示されている。

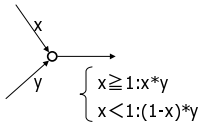


図 2.4: 演算ノード使用例

- アーク

  アークは変数ノードおよび演算ノード間をリンクし、値を一方向に伝播するために使用する要素であり、矢印で表記される。リンク先が演算ノードの場合は値をそのまま伝播する。リンク先が変数ノードの場合は、3 種類のアークが定義され、アークに沿って “＋” もしくは “－” が表記さ

れる場合は、リンク先の変数ノードの値に伝播された値を加算もしくは減算する。アークに沿って何も表記がない場合は、変数ノードが持つ値を伝播されてきた値に書き換える操作を行う。表 3.2 にそれぞれのアークの表記法を示し、図 2.5 にアークの使用例を示す。本使用例において、X および Y と表記された変数ノードはそれぞれ x および y の値を演算ノードへ伝播する。演算ノードから Z と表記された変数ノードへのアークは減算アークであり、Z と表記された変数ノードが z の値を保持していた場合、Z が保持する値は $z = z - x * y$ と計算される。

表 3.2: アーク表記法

アーク表記	種類
+	加算
−	減算
	書き換え

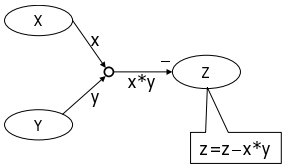


図 2.5: アーク使用例

2.2.2 シミュレーションモデル記述例

前節で説明したシミュレーションモデルの構成要素を用いたシミュレーションモデルの記述例を示す。図 2.6 のシミュレーションモデルは店舗で販売されている商品に対する消費者の購買行動をモデル化したものである。エージェントは店舗および顧客である。店舗エージェントは売値、売上数、在庫数、在庫補充、売上、収支を変数として持つ。顧客エージェントは消費度、消費の早さを持つ。まず、店舗は商品の売値を決定する。顧客は消費の早さによって消費度が減少し、消費度が一定の値を下回ると商品を購入するというイベントを発生させる。このとき、顧客は複数の店舗が示す売値の中から最も安価な店舗を選択して購入する。商品を購入すると顧客の消費度が一定量増え、同時に店舗の売上数が増え、在庫数が減少する。在庫数が一定の値を下回った場合には在庫補充を行う。店舗の売上は売値および売上数によって計算され、店舗の収支は売上および在庫補充にかかった費用によって計算される。

このように前節で述べた構成要素を用いてのモデル記述が可能である。

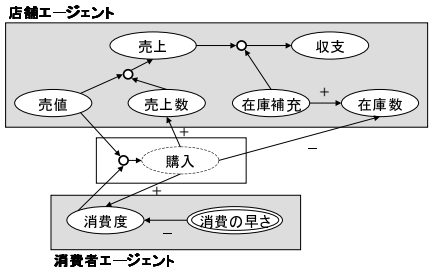


図 2.6: 商品購買のシミュレーションモデル

2.2.3 処理順序設定支援方法

処理順序設定支援について説明する。ここで、設定の必要のある項目は変数更新ルールの処理タイミングおよび演算の処理タイミングの 2 点である。ここで、変数更新ルールとは変数ノードが他の変数を参照せずに自律的に値の更新を行うルールのことであり、前述の商品購買モデルを例に挙げると、“売上数を 0 にリセットする”が変数更新ルールの 1 つである。これらの設定項目数はモデルの規模が大きくなるほど処理順序の組合せは膨大になり、分析者にかかる負荷が大きくなってしまう。提案する処理順序設定支援方法は、モデル構造に着目し、分析者が指定するイベント、出力ノードを用いて設定項目数の削減を図る。

図 2.7 に示す商品購買モデルを用いて本支援を説明する。分析者は対象問題の中心となるイベントおよび出力を指定する。ここで指定されたイベント内の処理が中心の処理となる。続いて、モデル構造を用いて、指定イベントへ影響を与える部分および受ける部分に分けられ、それぞれが指定イベント前に処理される部分 (売値、消費度など) および指定イベント後に処理される部分 (収支、在庫数など) として設定される。また、演算の処理タイミングについては、それぞれの場所において、指定出力のノードからの距離を用い、遠い演算ノードから演算が処理されるように設定される。距離とはアークを矢印の逆方向に辿ったときのアークの本数とする。本設定支援方法をがない場合とある場合の分析者の設定項目数を比較すると 10 個から 2 個に削減することができ、分析者にかかる負担を軽減することができる。

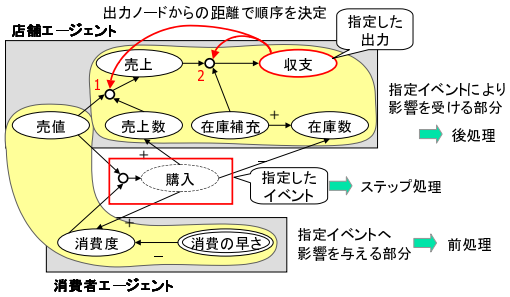


図 2.7: 設定支援方法

本モデルのような単純なモデルではなく、影響のフィードバックが起こったり影響のパスが分岐するモデルでは、上記の方法で自動設定することができず、複数の候補が考えられる場合はその部分を分析者に提示し、分析者により設定してもらう。しかし、本支援がない場合と比べると、設定個数が減少し、設定できなかった設定項目も本支援により決定される一部の設定に対して、前に発生するか後に発生するかを設定するのみであるため、設定が容易になる。

2.3 プログラムソースジェネレータ

MAS モデルエディタ上で作成されたモデルデータを持つ XML データを用いてプログラムソースを生成する。ここでは、前節で説明した商品購買モデルの XML データを用いた MAS のプログラムソース生成について説明する。

図 2.8 にエージェント、変数の宣言部分、およびグローバル変数の初期化部分、さらに、図 2.9 に universe およびエージェント記述部分の生成方法を示す。

エージェント、変数の宣言部分、およびグローバル変数の初期化部分では artisoc の記述ルールに従って変数名や初期値などを XML データから該当部分を抽出して記述する。

universe 部分ではステップの前処理 (Univ_Step_Begin) および後処理 (Univ_Step_End) を記述する。演算は変数の値を用いた計算を行うため、変数の更新ルールを記述した後に演算を記述する必要がある。そのため、XML データの rule タグおよび formula タグの場所を参照し、それぞれの場所において、変数の更新ルールの記述、演算の記述の順で記述する。また、複数の演算を記述

する場合は記述順を formula タグの order 属性の順に従って記述する。

更新ルールの記述を行った変数ノード、または演算を行った結果変化する変数ノードが演算ノードを介せずに他の変数ノードへ直接アークによってリンクされている場合、上述の例においては “在庫補充”→ “在庫数”部分のような場合は、その記述を行った直後に、変数ノード間にリンクするアークの種類に応じた演算を記述する。

エージェントの記述部分ではまず、エージェントが持つ変数の初期化部分 (Agent_Init) に、エージェントが持つ変数ノードの初期値を XML データから抽出して記述する。artisoc が定義するステップ処理は実行するシミュレーションのメインの処理であり、上述の例では購入の決定というイベントがメインの処理となる。記述方法は前後処理と同様に行い、変数更新ルールおよび演算の記述方法、記述した演算により直接の影響を受ける変数ノードの変化の記述を行う。また、エージェントの記述は定義したエージェントの数だけ記述する。

2.4 プログラム開発環境ユーザインターフェース

図 2.10 にプログラム開発環境のインターフェースを示す。インターフェースは上部のメニューバー、メニューバー下部のシミュレーションモデル作成のための構成要素メニューバー、さらに下部のモデル描画領域の 3 つの部分から成る。

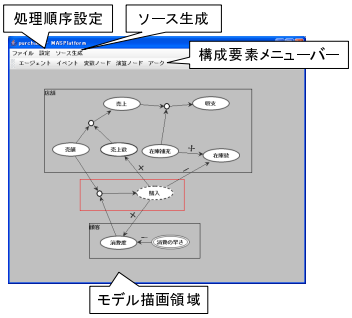


図 2.10: プログラム開発環境インターフェース

分析者は、構成要素メニューバーから作成する要素を選択し、モデル描画領域上に作成する。モデル作成後、分析者はメニューバー内の “設定”から処理順序の設定を行い、モデル作成および処理順序設定を終えた後に “ソース生成”を選択し、プログラムソースが自動生成される。

3 評価実験

3.1 モデル記述および設定支援の評価

提案するプログラム開発環境が提供するシミュレーションモデル記述および処理順序設定支援の評価を行う。モデル記述に関しては、様々な対象に対してモデルが記述可能であるかどうかが評

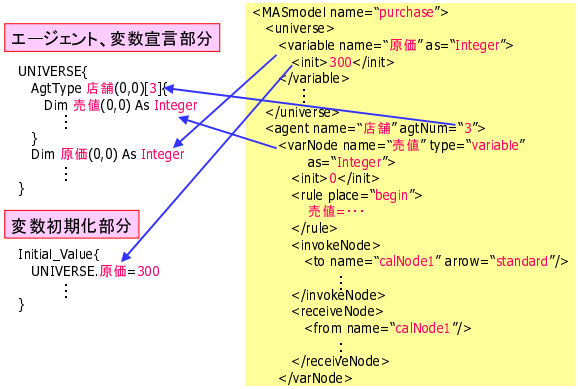


図 2.8: エージェント・変数宣言および変数初期化部分の生成

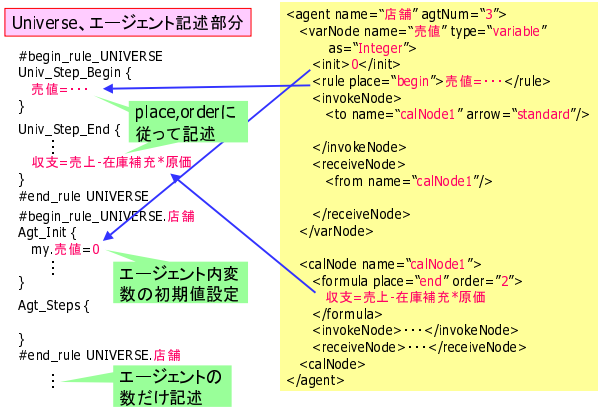


図 2.9: Universe・エージェント記述部分の生成

価となり、設定支援に関しては、分析者が設定する項目数を削減することができるかどうかが評価となる。

図 3.1 に示すように、ネットオークションのユーザおよび商品の振る舞いを表現するモデルが記述可能である。本モデルの場合、分析者の行う処理順序設定項目数は 9 個から 3 個に削減することができた。

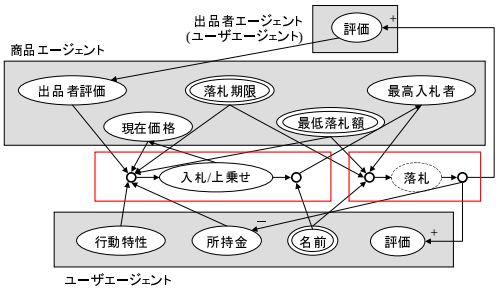


図 3.1: ネットオークションモデル

同様にして、図 3.2 に示すように、顧客および企業による月額課金事業 [12] の振る舞いを表現するモデル記述が可能である。本モデルの場合、設定項目数を 21 個から 3 個に削減することができた。

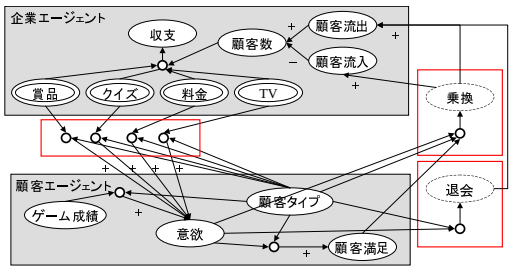


図 3.2: 月額課金事業モデル

以上から、プログラム開発環境が提供するモデル記述を用いると様々な対象をモデルとして記述することができ、処理順序設定支援を用いると分析者にかかる負荷を軽減することができると示すことができた。

3.2 プログラム開発環境の評価

提案するプログラム開発環境の評価を行うために、プログラムソース作成作業をプログラム開発環境を使用しないで行う場合を比較対象と、プログラムソース作成にかかる時間の比較を行った。

実験には MAS に関する知識はなく、Java や C 言語などのプログラミング言語を用いたプログラム作成経験のある被験者 2 名に、以下に示す対象問題および 2 つの方法でシミュレーションモデルの作成およびプログラムソースの作成をしてもらった。

- 対象問題
  - 2 種類のエージェント
  - 2 つのイベント
  - それぞれのエージェントで属性数は 7 個および 5 個
- プログラムソース作成方法

方法 1 被験者が自由にシミュレーションモデルを作成し、テキストエディタを使用してプログラムソースを作成

方法 2 本研究で提案するプログラム開発環境を使用してシミュレーションモデルを作成し、プログラムソースを自動生成

さらに、上記の対象問題に一方のエージェントに属性を 1 つ追加するという問題の変更を行い、同一の被験者 2 人に対してモデルの修正およびプログラムソースの変更の作業をしてもらった。

シミュレーションモデルの作成およびプログラムソースの作成を進めるにつれて対象問題に対する理解が深まる。そこで、2 人の被験者には、上記プログラムソース作成方法の方法 1 で行った後に方法 2 で行ってもらう被験者 (以下、被験者 A) と、方法 2 で行った後に方法 1 で行ってもらう被験者 (以下、被験者 B) に分けて実験を行った。

3.3 実験結果および考察

被験者 A および B にそれぞれ 2 つの方法で作成されたプログラムソースは一部異なる記述で処理が書かれた部分があったが、全体の処理順序や結果を見ると、いずれも同一の意味を持つ処理および結果を出力するプログラムソースが作成された。表 3.1 に、2 人の被験者および 2 つの方法によるプログラムソース作成時間および条件追加によるプログラムソース変更時間の比較を示す。

表 3.1: 実験結果

	テキストエディタ		プログラム開発環境	
	作成時間	変更時間	作成時間	変更時間
被験者 A	320 分	11 分	75 分	3 分
被験者 B	310 分	16 分	127 分	9 分

この結果に示されるように、2 人の被験者とも、提案したプログラム開発環境を用いてプログラムソースを作成した方が大幅な時間の短縮になった。さらに、テキストエディタによるプログラム作成・変更を実験後半に行った被験者 B に注目すると、問題に対する理解が深まっているにもかかわらず、プログラム開発環境を用いた方が短時間で実験を終了している。以上のことから、本研究で提案したプログラム開発環境は、MAS のプログラムソース開発を行う際に有用であることを示すことができた。

4 結論

MAS ツールを用いたプログラムソース開発の際に分析者にかかる負荷を軽減するためのシミュレーションプログラム開発環境を提案した。提案したプログラム開発環境は、シミュレーションモデルをグラフ構造で表現することとし、モデル内のエージェントおよびイベントを、“変数ノード”、“演算ノード”および “アーク”の 3 要素を用いた有向グラフで表現するよう定義した。さらに、プログラムソース生成の際に必要な、分析者が行う処理順序の設定にかかる負荷を軽減するための処理順序設定支援の機能を持っている。

定義したモデル記述法が様々な対象を記述可能であることを示し、処理順序設定支援により全てのモデルに対して設定項目数を削減できたことを示した。さらに、2 人の被験者によるプログラム開発環境およびテキストエディタを用いたプログラム作成・変更の時間比較を行い、両被験者ともプログラム開発環境によるプログラム作成・変更の方が短時間で終了でき、プログラム開発環境の有用性を確認した。

謝辞

本研究に際し、MAS ツールとして artisoc を無償提供頂きました (株) 構造計画研究所に心より感謝致します。

参考文献

[1] 北中英明: “複雑系マーケティング入門 〜マルチエージェント・シミュレーションによるマーケティング〜,” 共立出版 (2005).

[2] ロバートアクセルロッド, マイケル D. コーエン: “複雑系組織論 多様性・相互作用・淘汰のメカニズム,” 高木晴夫, 寺野隆雄　監訳, ダイヤモンド社 (2003).

[3] 大内東, 川村秀憲, 山本雅人: “マルチエージェントシステムの基礎と応用—複雑系工学の計算パラダイム,” コロナ社 (2002).

[4] R. Axelrod: “The Convergence and Stability of Cultures: LocalConvergence and Global Plarization.” Working Paper 95-03-028. Santa Fe,N.M.: Santa Fe Institute (1995).

[5] N. Minar, R. Burkhart, C. Langton, and M. Askenazi: “The Swarm simulation system: A toolkit for building multi-agent simulations,” Working Paper 96-06-042, Santa Fe Institute, Santa Fe.(1996).

[6] M.J. North, T.R. Howe, N.T. Collier, and R.J. Vos: “The Repast Simphony Runtime System,” in *Proc. of Agent 2005 Conference on Generative Social Processes, Models, and Mechanisms* (2005).

[7] “StarLogo on the Web,” http://education.mit.edu/starlogo/

[8] “MAS コミュニティー,” http://mas.kke.co.jp/index.php.

[9] 秋吉政徳, 木村慶史, 薦田憲久: “業務モデルをビルディングブロックとする事業シミュレーション方式,” 電気学会 情報システム研究会, IS-06-19, pp.27-30 (2006).

[10] 木村慶史, 秋吉政徳, 大礒洋明, 薦田憲久: “マルチエージェントシミュレーションによる事業シナリオ評価手法の検討,” 電気学会 情報システム研究会, IS-06-31, pp.31-34 (2006).

[11] 山影進, 服部正太: “コンピュータの中の人口社会,” 共立出版 (2002).

[12] S. Montananont, 正野勇嗣, 平松綾子, 鈴木亨, 大礒洋明: “構造方程式モデリングを用いた携帯コンテンツユーザの退会意思分析,” 電気学会 情報システム研究会, IS-04-54, pp.25-30 (2004).