共生ゴールエージェントの交渉と再構成による要求変化への対応

Requirement Change Handling by Negotiation and Reconstruction of Symbiotec Goal Agents

飯島 正 †
Tadashi IIJIMA†
† 慶應義塾大学　理工学部
†Faculty of Science and Technology, Keio Univ.

要旨

情報システムへ求められる要求は，時間の経過にしたがって変化しうる．そのため，要求の変化に対して迅速かつ低コストに追従できることが望ましい．そこで，要求に対応するゴールと，それを満たす機能とを，それぞれ自律的なエージェントで表現し，いわゆるマルチエージェントシステムとして，情報システムを柔軟かつ自己組織的に構成するアーキテクチャを提唱する．本報告では，個々のエージェントは，オブジェクト指向ペトリネットに基づいて振る舞いを規定し，エージェント間協調プロトコルに基づいたメッセージ交換によって，交渉と再構成を行うアーキテクチャを想定して，今後そうしたアーキテクチャの実現に取り組むにあたっての予備的な検討を行う．

1. はじめに

情報システムへ求められる要求は，時間の経過にしたがって変化しうる．しかし要求が変化するたびに，人手で再設計と再実装を繰り返していては，時間とコストがかかりすぎてしまう．そのため，要求の変化に対して迅速かつ低コストで追従できる性質を予め付与しておくことができればそれに越したことはない．

そこで，要求に対応するゴールと，それを満たす機能とを，それぞれ自律的なエージェントで表現し，いわゆるマルチエージェントシステムとして，情報システムを柔軟かつ自己組織的に構成するアーキテクチャを提唱する．ゴールエージェントは一つ一つが何らかの機能要求に対応しており，さらにその要求を満たす機能エージェントと対応付けられている．同じ機能要求に対して，必要な資源の量や性能等の非機能的品質が異なる「ゴール&機能」対が複数存在する．それら「ゴール&機能」対の間では，事前に定義されたエージェント協調プロトコルに基づいてゴールエージェント間で交渉することにより，資源割り当ての調整を行い，限られた資源の範囲で機能エージェントが要求を満たすように振る舞う．

個々のエージェントは，オブジェクト指向ペトリネットに基づいて振る舞いを規定するが，エージェント間協調プロトコルも一つのオブジェクト指向ペトリネットとして表現される．それらが相互に同期的に相互作用することで資源割り当ての交渉を行い，情報システムの中で資源を割り当てられて機能する機能エージェントが選定される．

もっとも完全に自動的に再構成を行うシステムにおいて，想定外の思わぬ不具合を混入させてしまうことが起こってはならない．必要な検証項目のリストを常に保守しておき，モデル検査等の動作検証を通過した場合にのみ，その再構成を有効化するような機能も含めておくことが望ましい．また，自動的な再構成を行うということから，こうしたシステムでは，そのシステム自身がどのように動作するかについて，利用者へ説明する機能も必要である．

本報告は，このようなアーキテクチャの実現に取り組むにあたり検討しておくべき項目を整理することを目的とする．まずは，具体的なアーキテクチャを想定した上で，どのような利点欠点があるか，どのような機能を持たせることが必要かという点について論じる．

2. アーキテクチャの概要

情報システムを自律的に再構成可能なものとするためには，情報システムを構成するモデルを操作可能なものとしなければならない．すなわち，モデル自体が First-Class Object であるようなリフレクティブ (Reflective) 性を持つ必要がある．

そこで情報システムを，ゴールエージェントと機能エージェントの組で表現し，構成するエージェント群の選択と，相互作用（同期）によって全体の再構成を可能とする．機能エージェントは，さらにサブゴールのゴールエージェントと機能エージェントの組で表現される．機能エージェントは，それ自体

がゴールエージェントと機能エージェントの組となっており，再帰的に階層構造を形成する (図 1)．ゴールエージェントだけに着目すると，AND/OR 分割によるゴール–サブゴール分解が繰り返されることで木構造（ゴール木）を形成することになる．これによって，その情報システムの機能全体を木構造上に表現することができる．

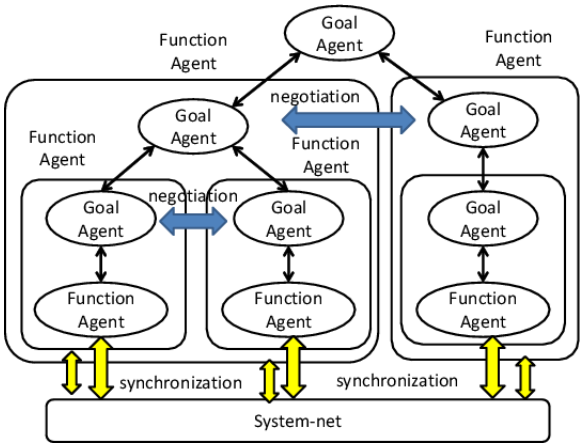

図 1: ゴールエージェントと機能エージェント

ゴールエージェントは，その階層構造に沿って，相互に交渉し，どのゴールを優先して実現するか，どの機能エージェントにどれくらいの資源を割り当てるかを調整する．その交渉に用いる協調プロトコルを共生プロトコル (Symbiotec Protocol) とよぶ．共生プロトコルを構成する基本的な協調プロトコルとして，現時点では契約ネットプロトコル (CNP; Contract Net Protocol) [2] と想定している．トップゴールから順に（葉の方向へ），AND 結合しているサブゴール群に向かっては実現を求められている機能を選択するための CFP(Call-for-Proposal) を送信する．一方，OR 結合しているサブゴール群に向かっては，機能は同等なので，そこで必要となる資源について選択するための CFP(Call-for-Proposal) を送信する．いずれのサブゴールも，プロトコルのイニシエータに対して入札を行い，順次，落札しながら，ゴール木を遡っていき，トップゴールに到達するまで繰り返す．落札基準は，交渉プロトコルのイニシエータとなるゴールエージェントに任せられることになる．

落札されたゴールエージェントは，その対応する機能エージェントを活性化させる．その結果，活性化された機能エージェントによって，情報システムが再構成されることになる．各機能エージェントの振る舞いは，それぞれオブジェクト指向ペトリネットで規定されたワークフロー（プロセス）として規定される．ここのプロセスは，次節で述べるように，トランジッションの発火と共に，対応する対応する Java(ないし Scala) のメソッドが起動されることとなる．

共生プロトコルを含む協調プロトコルも，次節で述べるオブジェクト指向ペトリネットとして与えられる．エージェント間の協調的な振る舞いを，同期関係によって定義することができる．

情報システムを再構成するトリガーは，要求の変化として与えられる．要求の変化は，ゴール木の上で，関連するゴールエージェントを含む部分木に対して，作用し，その部分ゴール木のルートを起点として，再度，資源割り当てのための共生プロトコルを開始することで自己組織的な再構成を行う．ここで，本報告では規定していないが，要求ないしゴールを表現するための言語体系が必要である．要求は業務ドメイン知識に依存しているため，ドメインごとに DSL(Domain Specific Language; ドメイン固有言語ないしドメイン特化言語) を構築することを検討中である．

情報システム全体は，次節で述べるシステム・ネットで統合される．図 1 ではシステム・ネットは階層化されていないが，実際にはゴール木と対称的な階層構造を形成してもよい．

3. オブジェクト指向ペトリネット

3.1. オブジェクト指向ペトリネットとは

本研究では，エージェントの振る舞い表現ならびに協調プロトコルの表現としてオブジェクト指向ペトリネットを用いている．オブジェクト指向ペトリネットとはオブジェクト指向概念に基づいてモジュール性を取り入れたペトリネットの拡張モデルであり，多くのものが提案されている．ここでは，nets-in-nets 意味論に基づく参照ネット (Reference Net) [1] の考え方を採用したオブジェクト指向ペトリネットを採用している．

参照ネットにおいては，トークンには 2 通りある．一つは，通常の P/T ネット (Place/Transition ネッ

ト) におけるトークンである単純トークン (Black Token) とよぶ．プレースに存在する単純トークンは，トークン数で示す．もう一つは，別のサブシステムを表現するサブネットへの参照 (reference) に相当する参照トークン (Reference Token) である．

プレースは，単純トークン用の単純プレースと，参照トークン用の参照プレースに分類できる．

- 単純トークン (Black Token)[1]
- 参照トークン (Reference Token)

ここで，参照トークンが参照しているサブネットは，オブジェクト・ネットと呼ばれ，一つのオブジェクトとしてのアイデンティティをもつ．参照トークンも，単純トークンと同様，ネットワーク中を遷移することができるが，一つの参照トークンがトランジションの発火によってコピーされることもある．その際には，あくまで共通のオブジェクトネットへの参照がコピーされる点に注意されたい．

後述する (図 2 左) ように，各トランジションの発火の際には，そのトランジションと同期する (interaction 関係にある) ペトリネットにおいては，同名のメソッドが実行される．すなわち，そうした同期関係 (interaction 関係にある) ペトリネットが実行主体 (アクター) として位置づけられることになる．受け渡しされるパラメータは，基本的に，そのトランジションへの入力アーク上の weight で表現される．それに加えて，同期関係にある他のネットから受け渡される参照が付け加えられる．それらは，?(入力パラメータ) ならびに !(出力パラメータ) として受け渡される．

3.2. オブジェクト指向ペトリネット OPeN

以下では，具体的なオブジェクト指向ペトリネットのモデルとして，筆者の主宰する研究室で継続的に開発を行ってきたオブジェクト指向ペトリネット OPeN ファミリー (the Object-oriented Petri Net family) の中でも，特に人と人の連携して行う協調作業や，SOA におけるサービス間連携を記述するためのビジネスモデル記述 (ワークフロー) に特化した OPeN/WF を用いている．

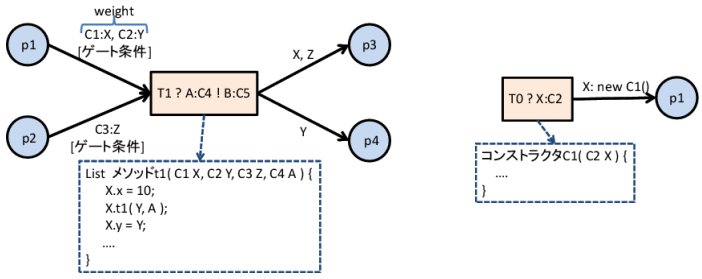

図 2: OPeN におけるアークとトランジション

OPeN では，個々のオブジェクト・ネットは，クラス記述から生成されるインスタンスに相当する．クラス記述は，オブジェクトを定義するものであり，本稿執筆時点では，opn(もしくは従来からの慣習から pnmlx) という拡張子を持つ XML ファイルとして記述する．クラスが定義するオブジェクト・ネットは，そのオブジェクトの振る舞い (ライフサイクル) を記述する一つのペトリネット，そのオブジェクトが持つプロパティ(インスタンス変数) 群と，ライフサイクルを遷移する際に使われるメソッド群である．メソッドは，主に，トランジションの発火条件であるゲート条件の記述や，トランジション発火時に起動されてプロパティの値の更新に使われる．図 2 に OPeN におけるアークとトランジションのアノテーションを示す．

トランジション t_1 の発火によって，同じ名前をもつメソッド t1 が起動される[2]．現時点のプロトタイプ仕様では，プロパティの型やメソッドの定義のための言語仕様は，厳密には定義していないが，図 2 に準拠し，基本的なリフレクション機能を備えた一般的なオブジェクト指向言語 (たとえば Java や Scala) のクラス定義 (ネットと同名のクラス定義) に外部化しており，相互に対応付けている．

複数のオブジェクト・ネットから構成されるシステムは階層的に表現されるが，これには，プロジェクトという単位を用いる．しかし，ここでは，まず分かり易さを優先して多階層の構造ではなく，2 階層の構造に限定して，その意味論を概念的に説明する．これは，[1] における EOS(初等オブジェクトシステム) に相当する．2 階層の構造では，全体的な振る舞いを統制するオブジェクト (ペトリネット) と，それに規定された振る舞いを行うサブオブジェクト (ペトリネット) 群から構成される．ここでは，全体

[1]慣習的にブラック・トークンと呼ばれるが，ここでは分かり易さのために単純トークンと呼ぶ

[2]後述するオブジェクト指向ペトリネットの発火則「Interaction(相互作用)」により，t_1 と同期するトランジション t_{s1} が，このペトリネットのサブネット中に存在し，かつそれが発火可能であれば，その t_{s1} も同時に発火する．t_{s1} が発火可能でなければ t_1 も発火できない．トランジションの入力アークがメソッドの入力パラメータに対応する．

的な統制を記述する方を EOS の慣習に従いシステム・ネットと呼び，統制されたサブオブジェクトの側をオブジェクト・ネットと呼んで説明することにする[3].

システム・ネットとオブジェクト・ネットの間では，一部のトランジションの発火が同期的に行われる．システム・ネットとオブジェクト・ネットの同期関係は，双方のトランジションの対の集合として表現される．システム・ネットにおいて，あるトランジション T が発火するためには，以下の三条件が成り立たねばならない[4].

(a) システム・ネットにおいて，単純トークン B_i に関して発火条件を満たしていること，
(b) システム・ネットにおいて，参照トークン R_j に関して発火条件を満たしていること，
(c) T の発火に寄与している参照トークン R_j が参照しているオブジェクト・ネット中で，T と同期関係にあるトランジションが発火可能であること．

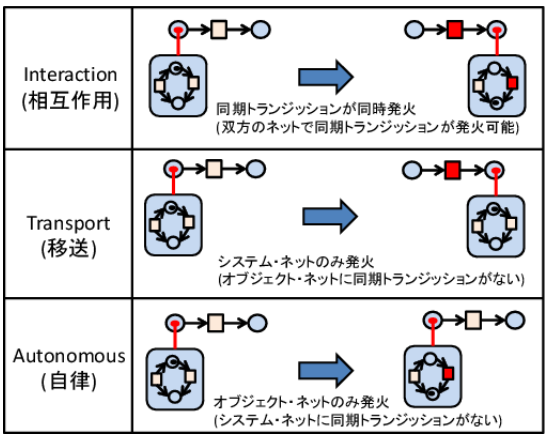

図 3: トランジションの発火則

こうした発火のメカニズムから，nets-within-nets 意味論 (ないし参照意味論)，およびその拡張に基づくオブジェクト指向ペトリネットにおいては，(a) interaction(相互作用), (b) transport(移送), (c) autonomous(自律)，という 3 通りの発火則が考えられる．まず，システム・ネットとオブジェクト・ネットが同期して遷移するタイプであり，interaction(相互作用) と呼ばれる．次にシステム・ネットの発火に際し同期して発火するオブジェクト・ネットのないタイプがあり，transport(移送) と呼ばれる．これは，オブジェクト・ネット内部には何の状態遷移も起こっていないまま，システム・ネット上にある参照トークンが移動する様子が，モバイル・エージェントの移動に対応づけられることから名づけられている．最後に，システム・ネットとは独立に，オブジェクト・ネット内のトランジションが発火するものであり，autonomous(自律) と呼ばれる．その呼称は，システム・ネットに制御されることなく，オブジェクト・ネット内部で自律的にトークンの遷移（すなわちそのオブジェクト・ネットの状態遷移）が引き起こされていることに由来する (図 3).

4. 自動的な再構成に伴う問題点と求められる機能

現時点では，第 2 節で示した基本的なシステム再構成機能に加えて，情報システムの信頼性を維持するためには，以下の 2 つの機能が必要であると認識している．

(1) 検証機能

(2) 説明機能

検証機能は，要求変化に応じた適応的再構成によって想定外の思わぬ不具合を混入させてしまわないようにするためである．また説明機能は，利用者が信頼してそのシステムを使い続けるためには，その時点での機能を理解し納得することがであることから重要である．以下ではそのそれぞれについて検討を加える．

[3]階層的なシステムの場合には，ここでいうオブジェクト・ネットもまた，より下位のサブペトリネットからみれば相対的にシステム・ネットに相当する．OPeN は EOS の概念を多階層に拡張しているが，混乱のない限り，最上位ネットに限らず，隣り合う階層のネット間の関係を，システムネットとオブジェクト・ネットというように称することとする

[4]但し，OPeN の場合には，二階層の EOS ではなく，多階層を許容しているので，条件 (c) は再帰的に，より下位層にあるオブジェクト・ネットに対して適用されていくことになる．

4.1. 検証機能

機能エージェントの再構成を自動的に行う場合，想定外の思わぬ不具合を混入させてしまうことが起こってはならない．したがって，再構成したシステムを有効化する前には，毎回，全体的な整合性を検査する検証作業が必要である．必要な検証項目のリストをゴール木に対応付けて常に保守しておき，再構成のたびにモデル検査等の動作検証を行う．検証を通過した場合にのみ，その再構成を有効化するものとする．現時点では，ペトリネットの表現力を有限状態モデルのレベルに限定してモデル検査に適用することを検討中である．

問題は，要求を提示する利用者にも理解できる言語体系で検証項目を与えることができるかどうかにあり，そのための DSL の構築も合わせて検討している．

4.2. 説明機能

自動的な再構成を行うということから，こうしたシステムでは，そのシステム自身がどのように動作するかについて，利用者へ説明する機能も必要である．ゴール木から有効化された機能エージェントに関連する部分木を抽出し，それを元に，説明を生成する．この説明も，要求を提示する利用者にも理解できる言語体系で生成できなければならない．これには要求（ゴール）を記述する DSL と同じ語彙体系の利用を検討している

5. おわりに

時間の経過にしたがって要求が変化する情報システムに対して，ゴールエージェントと機能エージェントの対から構成されるマルチエージェントアーキテクチャによって，変化への追従を自動化することを提唱した．本報告では，将来的な実現に向けて検討中の，こうしたアーキテクチャに求められる要件項目に関して報告した．

参考文献

[1] Rüdiger Valk: “Object Petri nets? Using the nets-within-nets paradigm, ” LNCS 3098, pp.819–848, Spriner, 2004.
[2] 本位田真一，飯島正，大須賀昭彦: “エージェント技術,” 共立出版, 1999.