グループウェア 36－4
（2000．5．25）

契約と適合化
--- エージェント･モデルに基づく ワークフロー管理とコンポーネント結合 ---

飯島 正，山本 喜一，土居 範久（慶応義塾大学理工学部）

筆者の提案している共生・寄生モデルに基づくソフトウェアエージェントに契約の概念を導入し，ワークフロー管理とコンポーネント結合に適用する構想を紹介する．

共生・寄生モデルは，移動エージェントを含む，拡張可能なソフトウェアエージェントのための概念モデルである．このモデルでは，エージェントは他のエージェントに寄生することができ，それにより移動行為を説明したり，動的な機能獲得や，監視エージェントの派遣といった適合化のパターンを表現することができる．

このモデルに従うエージェントを，ワークフローとその管理のモデリングにおける

1) プロセス（アクティビティ），2) プロダクト（ドキュメント），3) パーティシパント

といった要素ををモデル化するために適用している．さらにワークフローにおいて随所に現れる「契約」の概念をコンポーネント結合に活用していく上での出発点とする．

Contract and Adaptation
--- Workflow Management and Component Binding based on Agent Model---

Tadashi Iijima, Yoshikazu Yamamoto and Norihisa Doi (Keio University)

This paper describes about some proposals to introduce "contract" concept into agent technologies based on the "Parasitic/Symbiotic Agent Model" which was proposed by the authors, and apply to workflow management and bindings among agents in general.

The model provides ability to organize dynamically and abstract a nested group of agents as a single agent. Such a nested group can be transported through the network in the same way as a single agent. The model also presents an abstract model of mobility of such agents.

In this paper, agents based on the model are treated as technologies to model some entities in workflow management, such as processes(activities), products(documents) and participants. In this paper "contract" concept is introduced by using the workflow management model.

1. はじめに

近年，システム構成要素の自律分散化を目指したソフトウェアエージェント技術[2]が注目されている．各エージェントは，ACL(Agent Communication Language)と呼ばれる言語による通信，プランニング，交渉といった機能の一部もしくは全部を備えることにより，互いに柔軟な結合をし合う．そうしたエージェント技術の一つとして，筆者が提案する共生・寄生エージェント・モデルがある．

同モデルは，システムを構成する構成要素（コンポーネント）としてエージェントを捉え，エージェントの集まり（マルチエージェント）によって，システムを構成することを前提として，そのエージェントに各種の自律性を与えておくことで，動的なシステム構成の変更発展を意図したモデルである．自律性の中には，動的な移動機能（モバイルエージェントの持つネットワーク上の移送機能）も含まれ，本モデルを特徴付けるような幾つかのアプリケーションを作成している．エージェントは基本的に寄生機能というコンポーネントの入れ子状の動的構成機能を有しており，ネットワークをまたがった寄生にはこのネットワーク移動機能が有効に働く．しかし，本モデルは決してモバイルエージェントに限定したものではなく，静止（常駐型）エージェントを含めて，ソフトウェアコンポーネント一般を想定したモデルである．

こうした動的なシステム構成の変更発展（再構成）機能を支えるのが協調機能である．環境の変化に対しシステムが「適合化」するためには，これまで結合していたコンポーネントと別のコンポーネントをつかう必要が生じるかもしれない．協調機能はコンポーネントの結合を柔軟にするメカニズムであり，コンポーネントの追加交換を従来の強結合モデルよりも容易にする．さらに，現在，「共生における双利性」を意識したコンポーネント間の検索／結合に関して，研究を進めている．その試みのうちの一つがコンポーネント結合における「契約」の概念の導入である．

「契約」の概念は，エージェントの間で形成されるサービス提供者（サービスプロバイダ；サーバ）と依頼者（プリンシパル；クライアント）で時限的に結ばれる関係である．但し，いわゆるクライアント／サーバ間の単なるリクエストと異なる点は，履行の「義務」，ないし，不履行時の「救済」を導

入している点である．これは，一般的に法律的な用語として使われる「契約」という概念[10]に対応している．

契約というキーワードを用いたコンポーネント結合の研究は過去にも幾つかある[15][16]，それらは本来の意味における契約とは程遠い．というのは，契約不履行に関するペナルティといった概念がないためである．たとえば，オブジェクト指向モデルにおける「契約による設計」は，リクエスト／リプライモデルにおいて，リクエスト内容を事前条件／事後条件によって明確に仕様化するというものであって，契約のプロセスがコンポーネント間結合に含まれているわけではない．また，協調エージェントにおける契約ネットプロトコルは，競争入札のプロセスを含んだコンポーネント間結合モデルであるが，契約の履行／不履行以前の契約相手の選定に重点が置かれたものであり，やはり，契約そのものを扱ったモデルにはなっていない．

この「契約」概念は，契約のワークフローをモデル化することならびに，契約モデルをワークフロー管理に適用してみること[1]を通してモデル化を進めており，その結果をもとに一般的なコンポーネント結合方式へ発展させるという手順で進めている．現在，一般的な「契約」のフローをエージェントのメタレベルの協調プロトコルとして整理して，コンポーネント間結合の基本モデルの一つとして取り入れている．本稿ではその一部を紹介する．

また，本稿では，そうした「契約」概念と共生・寄生エージェントモデルが目的の一つとしていた「適合化」の概念を紹介し，それを盛り込んだワークフロー管理モデルについて紹介する．

本稿の構成は以下の通りである．次章では[3][4]に基づいて，共生・寄生モデルの概略と「適合化」のパターンを紹介する．続く，第3章では，ワークフロー管理への応用として，ワークフローの各構成要素のエージェントによるモデル化[1]について述べる．第4章では「契約」概念の導入について述べる．

2. 共生・寄生エージェントモデル

2.1. モデルの概略

共生・寄生エージェントモデル[1][2][3][4]は，筆者の提案する可変エージェント(mutable agent)のための概念モデルである．このモデルに対して，その実装のためのプラットフォーム（エージェントシステム）のフレームワークとしてJava 言語により寄生エージェント・フレームワーク（PAF）[2]が作られており，その改良も続けられている．

同モデルは，アプリケーションを階層的に構成する部品としてエージェントを位置づけ，その動的で柔軟な再構成を「寄生」という概念で整理したものである．また，この「寄生」は，ネットワーク上で離れた位置にあるマシン中に存在するエージェントに対して行われるとき，移動エージェントの「移動」行為そのものもモデル化している．「移動」に際して，エージェントは，一般に，内部に階層的に寄生している寄生者エージェントを抱えたまま，他のエージェントに寄生することが許される．これによって，アプリケーションを構成するエージェント群の関係を保ったまま移動することができる．

共生・寄生モデルは，動的に階層構造を形成／再形成することによる「ソフトウェアの大規模化」や，開放ネットワーク環境に不可欠な「機能の動的獲得」等のための基本概念を提供するものである．また，このモデルは，それ自体は既に移動エージェントだけを対象としたものではない[4]が，移動エージェント(mobile agent)の概念的な基礎モデルともなっている．紙数の都合上，その詳細は[2]に譲り，[3][4]に基づいてその概略のみを与える．

共生・寄生モデルでは，エージェント間関係を構成する次のような能力を与えている．1)［寄生］エージェント（寄生者）が他のエージェント（宿主）の内部に寄生するという操作を中心に，2)［獲得・吸収］，3)［排出］，4)［脱出］．5)［合成・統合］，6)［分割・委譲］などがある．

こうしたエージェント間関係の操作機能以外に，エージェントは相互にデータ通信し合う能力を有しており，上記のエージェント間関係の操作に先立ち，関与するエージェントの間で交渉が行われて，合意に達した場合に限り，操作が行われる．

これらの能力は，それぞれ固有の目的と権限ならびに能力をもった複数のソフトウェア・エージェントが混在するネットワークにおいて，協調的な関係（共生，寄生）を動的に形成することに貢献する．

移動エージェントの概念自体もこの枠組の中で与えられる[4]．通常，移動エージェントのモデル化では，エージェントが存在するための場所（プレース）を，ネットワークで相互接続された各ホストマシン上に配置しておき，エージェントはその間を移動する．これを，共生・寄生モデルで説明すると以下のようになる：プレースはそれ自身がエージェントである．プレースに移動エージェントが滞在していることは，そのプレースに別のエージェントが寄生していることを意味する．この時，プレースは宿主エージェントと呼ばれる．エージェントの移動は，その宿主エージェントから出て，別のプレースにあらためて寄生することに相当する．

共生・寄生モデルが提供する「能力・機能の動的な獲得や交換」の機能は，こうした移動エージェントにとって，実用上，極めて有用であろうと考えている[2]．移動エージェントが移動先で新たな能力を獲得する必要が生じるという状況は，十分考えられる．特にセキュリティを考えた場合，エージェントMAは，あるプレースP1（本モデルではプレース自体がエージェントないし疑似エージェントとしてモデル化されるが）内では，A という能力を発揮できるとしても，別のプレース P2 に移動した場合に同じ能力 A を行使できるとは限らない．たとえば，そうした状況は，A という能力の行使に，そのプレース内で有効な authority(A)という権限の取得が必要であるような場合に起こりうる．たとえば，あるホストに設置されているデータベースへのアクセスを能力 A として考える．本モデルでは，そうした権限の委譲をも「能力の動的獲得」としてモデル化する．「能力の動的獲得」は，能力 A（もしくは権限 authority(A)）を持ったエージェント AA の寄生によって行われる．寄生は，移動エージェント MA と AA の間の交渉の結果として引き起こされる（もしくは交渉決裂して失敗する）．エージェントAAは，そのホストに

あるエージェント以外のサーバ(データベースサーバ)が提供するサービスのラッパー(wrapper)ないしメディエータ(mediator)であるが，エージェントMAのネットワーク移動に伴ってエージェントAAも移動する．宿主エージェントの移動は，それに寄生しているエージェントごと行われる．但し，移動を禁止するような指定をすることもでき，そのとき，エージェントAAはプロキシとして遠隔的なリクエストの仲介を行う．移動先に別のデータベースがあり，そのデータベースに対して同種のアクセスをするためには，そのデータベースのためのアクセス能力をまた新たに獲得することになる．

2.2. 適合化のためのパターン

共生・寄生エージェントモデルでは，コンポーネントの再構成によるシステムの適合化のパターンを収集整理している．現時点では，[2]で紹介している4つのパターン(部品エージェントの寄生による機能拡張)に加えて，さらに幾つかのパターンを収集している．その一部を紹介する．

まず，部品エージェントの寄生による機能拡張[2]パターンとしては，(1)現地調達，(2)待ち伏せ，(3)召喚，(4)派遣がある．

(1)現地調達と〈2〉の待ち伏せは典型的にはモバイルエージェントが移動先の環境の中で適切な部品を獲得するパターンである．エージェントの移動はそのエージェントを取り巻く環境の変化をもたらすので，その環境に適した部品をその環境中に置いて管理することで行く先々で的確な部品を獲得することができる．現地調達では，モバイルエージェント自体が積極的に部品を獲得するタイプであり，待ち伏せでは環境の側から積極的に部品を送りこむ者である．しかし，これらのパターンが適用できるは，いわゆるモバイルエージェントに限らない．移動ロボットや人間が携帯するPDA(携帯端末)なども(物理空間上の移動によって)環境の変化にさらされ，環境に応じた振る舞いが要求される．そうした振る舞いを環境に埋めこんで管理しておき，さらにそうした組み込み機器の資源〈メモリ〉不足を補うために，環境への出入りの際に交換するパターン[13]「環境に埋めこまれた振る舞い」と「ネットワークスワッピング」)も提案している．[13]は，振る舞いを表現するコンポーネントの結合方式として包摂アーキテクチャの層の動的な再構成も提案しているが，これについては先行する同種の研究Gaeaのあることが，その後，判明した[14].しかし，その動的包摂アーキテクチャの先行研究は，振る舞いを環境に埋め込んで管理するというものではない．

(3)召喚と(4)派遣は，部品エージェントの方をネットワークを介して送りこむパターンであり，それぞれ，プラガブルコンポーネントを，いわゆるダウンロードとアップロードによって拡張することに相当する．

その他，現在，整理収集中の機能拡張パターンとしては，各種仲介を目的としたものがある．これらは，前述の(1)-(4)を使った，より上位レベルのパターンである．それには(6)クッション，(7)プラグ，(8)イベントコーディネータ，〈9〉プロトコルマネージャ等がある．クッションは，移動してきたエージェントが使うシステムコールのAPIと環境のAPIの間のトランスレータであり，入れ子の中間層に位置して必要に応じて導入／交換される．(7)プラグはJiniと同じようにある種のプラグアンドプレイを実現するために，必要なドライバを交換するものである．〈8〉イベントコーディネータは，イベントに基づいたコンポーネント結合モデルの元で，イベント名をすげ替えたり伝播させることを目的としたコンポーネントを導入するパターンである．(9)は通信プロトコルを共有するための(通信内容を送るためではなく，プロトコルを共有するという点に注意)派遣エージェントであり，プロトコルの監視も行なう．

3. ワークフローとその管理のモデル

3.1.エージェントに基づくワークフロー管理モデル

ワークフロー管理にエージェントモデルを導入する目的は，より柔軟で利用者にとって有用な機能を備えたワークフロー管理システムの実現けでなく，ワークフロー定義者にとってもワークフローのモデリングを簡単化するような概念を提供することにある(つまり，ワークフロー管理のモデル化とワークフロー自体のモデル化の両方を目的とする)．

一般に，既存のワークフロー管理システム[7]においては，ワークフローはプロセス中心にあらかじめ定義される．そのモデル化では，プロセス定義データ(手順，制御フロー)といったメタデータと，実行主体(エンジン)，そして操作対象であるデータとが分離されている．プロセス中心ではなく，プロダクトやユーザ(パーティシパント)を中心としたワークフロー定義はあまり一般的ではない．また，そのため，ビューもプロセス中心に固定されることが多く，実行時に生成されるワークリストがユーザ(パーティシパント)中心のビューとしてみられる程度である．

本稿で述べるワークフロー管理モデルは，移動エージェントを含む可変エージェント(mutable agent)のための共生・寄生エージェントを用いて，ワークフロー管理に現れる諸要素をモデル化することを試みている[1]．その中心的な要素として具体的には，プロセス，プロダクト(ドキュメント)，パーティシパント(担当者)の三つをエージェントとしてモデル化する(図1)．特に，ワークフロー(その中心はプロセス)とプロダクト(ドキュメント類)は移動エージェントとしてモデル化する．ワークフロー(プロセス・エージェント)はそれを生成したユーザの「代理」としてあるタスクを遂行するエージェントであり，プロダクトエージェントはそれを生成もしくは修正したパーティシパントの「代理」として情報を伝えたり蓄積したりするエージェントである．共生・寄生モデルを用いることで，ネットワーク中を移動する移動エージェントであるプロセス・エージェントにプロダクト・エージェントを寄生させたり，プロダクト・エージェント自体を入れ子状に階層化したりということが可能になる．

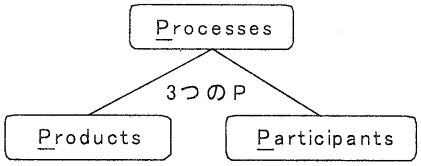

図1. エージェント化対象の三つのP

パーティシパントをエージェントとしてモデル化することは，従来よりしばしば行われている．WorkWeb System[6]でも人間のパーティシパントに対してはそうなっている．共生・寄生モデルでは人間のパーティシパントだけではなくアプリケーションソフトウェアも同様にエージェントとしてモデル化する．人間のパーティシパントの場合には個人の情報とタスクを管理するエージェントとして，アプリケーションソフトウェアをパーティシパントとみなしているときには，そのソフトウェアをラッピングするエージェントとなる．

3.2. ワークフロー構成要素のモデル化

ワークフロー管理とは，「ビジネスプロセスの一部ないし全部を，計算機システムを用いて自動化すること」を意味し，そのためのシステム（「ワークフロー製品」と呼ばれる）は既に多くのものが実用化されている[7]．ワークフローは「ビジネスプロセスの一部ないし全部の自動化」（WfMC）[8]と定義されるが，一般に，ワークフロー自体は，組織をまたがった文字通り「作業の流れ」と考えて差し支えない．その定義の中心はプロセスと呼ばれる「アクティビティの系列」である．但し，この系列は，逐次的なものに限らず，手続き的なプログラミング言語と同様に，条件分岐（合流），繰り返し，並行処理といった制御構造を持ちうる．

一般に，ネットワークを介して遂行されるタスクであるワークフロー（ビジネスプロセス）を表現するのに，移動エージェントを利用することは，自然なことのように見える．移動エージェントで追求されている自律制御，同期や待ち合わせ（合流）のような制御，遠隔ノードの監視やそれによって実現される大域的な一貫性の保守管理といった機能は，ワークフロー管理においても有効である

ワークフローには手順だけでなく，そのフローにおいて遂行される，各アクティビティの担当者や開始／終了の条件，関連するデータやアプリケーションプログラムなどが記述される．言い換えれば，ワークフローは，担当者間に情報を流していくルートと，情報が流れてきたときに担当者が遂行すべきアクションを規定しているとも言える．ここでいう担当者は，人間であることもあれば，アプリケーションソフトウェアということもありうる．[9]においては，アクティビティを遂行する担当者などを意味する役割と，その間を受け渡される電子伝票をオブジェクトとしてモデリングしている．

本稿のワークフロー管理モデルでは，ビジネスプロセスをモデリングするための構成要素として，プロセスとプロダクトとパーティシパントの3要素を考え，その3要素をすべて共生・寄生エージェント・モデルによってモデル化することを試みる[1]．

プロセスは，ワークフロー全体の振る舞いの記述でありアクティビティの系列である．プロダクトはワークフロー中で生成されるドキュメント類であり，典型的なビジネスプロセスでは伝票がこれにあたる．他に，生産活動を含むプロセスでは，その全ての成果物がこれに含まれる．パーティシパントは，いわゆるアクティビティの担当者であり，人間の場合とアプリケーションプログラムの場合がある．

3.3. ビジネスプロセスのモデル化

移動エージェントは，一般にその利用者の意図を代行して，ネットワーク中を移動しながら，その意図を満足させるタスクを遂行する．継続実行が可能なエージェントによって，ワークフロー（ビジネスプロセス）を表現することにする（ワークフロー・エージェントと呼ぶ）．フロー中には複数のアクティビティが順序付けられているが，これらもエージェントであり，ワークフローに寄生している存在である．イベントがワークフローの実行によって発火し，それが，より内側に寄生するサブエージェントに伝播して通知される．プロセス中のスレッドの分岐（並行処理）は，共生・寄生モデルに基づくエージェントの分割と合流によって表現することになる．イベントによって結合するモデルを利用することにより，寄生による機能追加／交換が柔軟になる（図2）．

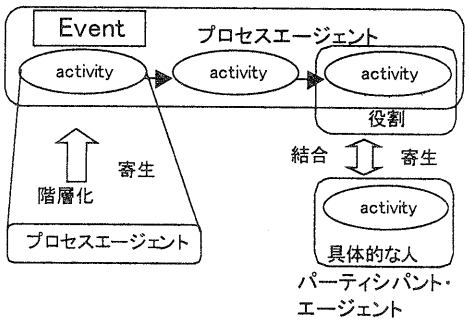

図2. プロセスとパーティシパント

更に，共生・寄生エージェント・モデルを用いることにより，プロセスの一部（サブプロセス）を当初から別のエージェントに記述しておいて，それをメインプロセスのエージェントに寄生させておくことも有用である．こうしたサブプロセスを意味するエージェントは，ワークフロー遂行中に追加したり，交換したりすることができる．ある程度は定型的な業務とはいえ，予め条件分岐を網羅しておくことが難しかったり，状況の変化に対応するケースも考えられる．そのためには，特定のサブプロセスエージェントを，初めから移動先のノードにおいてパーティシパント・エージェントから獲得するという適合化パターンを使うことができる（図2）．

より積極的な，共生・寄生モデルの利用法としては，ワークフロー・エージェントに対し，中間生成物であるプロダクト・エージェントを寄生させて，一緒に移動するというパターンがある．これは，永続的なプロダクトではなく，伝票の

ような一時的なプロダクトに対して有効である.

プロセスを表現するワークフロー・エージェントは, そのワークフローの遂行に責任を持つと同時に, そのプロセスの一部を改変する権限を持つ. このために, プランニング機能(とプランニングのための局所的な知識)を持ったプランニングサブエージェントを動的に獲得する(寄生させる)こともできる. すなわち, WorkWeb System[6]におけるBPTエージェントと同様に, 動的再計画をおこなう機能を付与する.

3.4. プロダクトのモデル化

ここでの典型的なプロダクトは, 伝票であり, それをエージェントとしてモデル化することは, [9]において電子伝票をオブジェクトとしてモデル化することに相当する. プロダクトに階層的な構造を持たせたり, 内容の一部の追加や交換を行なうことは, 寄生の概念に基づく「適合化」をもって説明できる. 期限付きの承認(承認印に相当するプロダクトエージェント)をエージェントとしてドキュメント(エージェント)に添付する(寄生させる)といったときに, 有用である.

3.5. パーティシパントのモデル化

アクティビティを実際に遂行するのは, 担当者(パーティシパント)である. 担当者が人間である場合には, 個人情報の管理やアクティビティの遂行を依頼/管理するワークリストハンドラ等の機能を備えた電子秘書のようなパーソナル・エージェントとなる. 担当者がアプリケーションソフトウェアである場合には, それをラッピングしたエージェントがこれにあたる.

プロセス(エージェント)とパーティシパント(エージェント)は互いに実行時にバインドされる. これは, プロセスエージェント中に寄生しているアクティビティ中で役割として抽象化されている要素が, パーティシパントエージェントのもつ役割と結合されてパーティシパントで具体化されることを意味する. その具体化は寄生によって行われる. プロセスエージェントに寄生しているアクティビティをパーティシパントエージェントにも同時に寄生させ, 両者で共有することによって, パーティシパント・エージェントの側から見ると実行すべきアクティビティを獲得することになる. 但し, そのアクティビティには実行の順序があり, それをワークフローエージェントが監視し順序制約の成立を保証する.

このメカニズムは, プロセス中心のビューだけでなく, パーティシパントを中心としたビューを形成する. あるパーティシパントに寄生している複数のアクティビティは, そのパーティシパントが遂行すべき「活動」の集合を意味する.

4. 「契約」概念の導入

4.1. 契約の概念

エージェント協調プロトコルとして「契約」ネットプロトコルが知られているが, 本モデルで導入する「契約」概念からいえば, これは「契約」ではない.

本モデルにおいて「契約」は,

契約＝合意形成＋{義務, 救済}

として図式化される. この図式は[10]における「契約の定義」に基づいている. 「契約ネットプロトコル」においては, このうちの「合意形成」以外は取り上げられていない. その「合意形成」も「交渉」を伴わない(「提案」を広い意味での交渉に含めることもできるが)ものである(多段階交渉プロトコルといった「交渉」を実現するバリエーションはありうる). ここで, 「合意形成」の結果, 一般には「申し出」と「承諾」からなる「約束」がなされる. 一旦, 「約束」がなされると履行に対する「義務」が発生し, その不履行に対しては「救済」が行われる.

共生・寄生エージェントモデルでは, この「契約」概念の3つの構成要素に関してそれぞれ3つのエージェントを与えている. 「合意形成」に関しては交渉に先立つプロトコル記述(これ自体がワークフローである)の交換を行うエージェント, 「義務」に関しては関連する「推論」に基づいて履行の「監視」とそれに続く「催促」ならびに「通達」を行うエージェント, 「救済」に関してはフローの「キャンセル」やパーティシパントへ「ペナルティ」や「賠償」を課したり「代替案」フローを提供するエージェントを配している. 最後のものは, 可能な範囲で再計画をおこなうためのエージェントと位置づけられる. 「義務」に関する推論のためには, 義務論理[11]の適用が有効と思われるが, これに関しては今後の課題である. これらのエージェントは対象に寄生し, その移動に伴って随行する.

4.2.契約の基本的なフロー

図 3 に基本的な契約のフローを示す(この表現は説明用のアドホックなものである). このフローをプロセスエージェントとして与えておく. そうすることで, そのエージェントを寄生操作により具体化/改変して, 実際のフローインスタンスを生成することができる.

ここで, 注意すべき事は, 検収のアクティビティと, それが不受理に終わった場合の救済のアクティビティを持つ点である. 契約のバリエーションとしては, 申し出(offer)を報酬の先払いで代用するようなケースもある(たとえば, 書籍購入の申込みと代金振り込みが同時に行われる場合). そうした場合には, 代金が不足していると, まず催促を試み, 失敗するかタイムアウトによって返金行為に移る. 返金行為は図では契約不履行の場合の依頼者救済行為として与えられている. こうした行為は, イベントによって起動するプロセスを内包したプロセスエージェントとして与えられ, 適宜, 追加することができる.

4.3. ワークフローにおける契約とコンポーネント結合

ワークフローにおいて「契約」は随所に現れる. 「物品売買」のような場合に「契約」が現れるのは当然だが, パーティシパントとワークフロー・エージェント(インスタンス)の間で行われるバインディングにおいても, この「契約」は含まれている. これは, 一般的な社内ワークフローでは暗黙的に成立している「雇用」契約に相当する. 現在, 依頼者(プリンシパル)とエージェント(代理人)の間にモニタリングとインセンティブ(報償)を導入することを進めている.

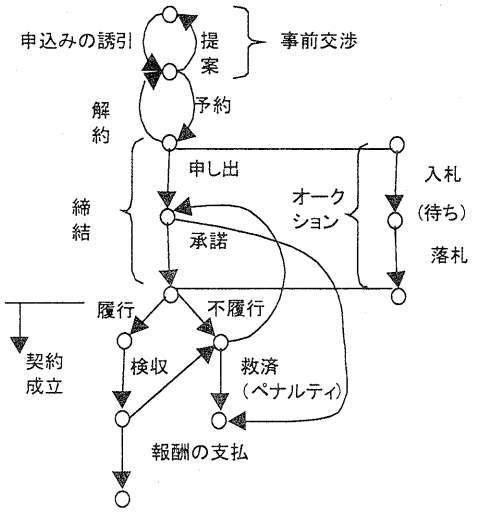

図3. 基本的な契約のフロー

5. 関連研究

ワークフロー管理を含む協調作業支援(CSCW)一般に関して，エージェントの概念に基づく多くの研究が既に見られ([5]など)，エージェント技術の持ついろいろな側面をワークフロー管理に取り入れている．

たとえばワークフロー管理システム WorkWeb System[6]は，エージェントの重要な特徴の一つである「動的」な性質を取り入れている．マルチエージェントによる動的な再計画，最適化を特徴とするものである．このシステムでは，各ユーザ，リソース，共有データサーバに対応してエージェントを割り当て，加えてワークフローのインスタンスに対してそれを管理するエージェントを割り当てる．ワークフローのプロセス定義は別途，Scheme 言語風のデータとして与えるので，このエージェントは，ワークフロー自体というよりワークフローエンジンをエージェント化したものである．それにより他のエージェントとコミュニケーションをとりながら(たとえば)例外の発生を検知し必要な再計画を行なうという機能強化を施したものといえる．

6. おわりに

本稿では，筆者の提案する共生・寄生エージェントモデルのワークフロー管理への適用に関して紹介した．プロセス(Process)，プロダクト(Product)，パーティシパント(Participant)の「三つの P」[12]の要素を寄生するエージェントとしてモデル化することに加え，「契約」概念を導入することで，より柔軟かつわかりやすいワークフロー管理モデルに向けてアプローチしている．「契約」に関する「推論」のための義務論理[11]の導入は今後の課題であるが，ワークフローの形式的な検証につながる重要な要素である．

また，本稿では触れていないが，通信等の諸事情からユービキタスコンピューティングが実現しえない現状では，モバイルコンピューティング(ノマディックコンピューティング)における利用者の位置移動に対する透過的なサポートも共生・寄生エージェントモデルに基づく移動エージェントがアプローチできると期待できる．

謝辞

義務論理に基づくワークフロー管理関連の参考論文の入手に関して NTT 情報流通プラットフォーム研究所の由良俊介氏にお世話になりました．この場を借りて感謝いたします．

参考文献

[1] 飯島，山本，土居:「共生・寄生エージェント・モデルのワークフロー管理への応用」，電子情報通信学会技術報告，人工知能と知識処理研究会(1998 年 12 月)．

[2] 本位田，飯島，大須賀:「エージェント技術」，共立出版(1999)．

[3] 飯島，山本，土居:「移動エージェントのための共生・寄生モデル」，第 55 回情報処理学会全国大会，平成 9 年後期(1997 年 9 月)

[4] 飯島，山本，土居:「拡張可能なエージェントのための共生・寄生モデル」，電子情報通信学会知能ソフトウェア工学研究会(1998 年 1 月)

[5] J.H.Connlly and E. A. Edmonds(Eds.): "CSCW and Artificial Intelligence'" Springer-Verlag (1994)

[6] 垂水，喜田，柳生，石黒:「エージェントによるワークフローの動的再計画」，情報処理学会論文誌，Vol.39, No.7 (1998)

[7] 戸田，飯島，速水，堀内:「ワークフロー － ビジネスプロセスの変革に向けて －」，日科技連(1998)

[8] WfMC (The Workflow Management Coalition) ホームページ，,http://www.aiim.org/wfmc/

[9] 稲本，佐藤，水野，片岡:「ワークフロー管理システム構築のための汎用的なオブジェクト指向モデル」，情報処理学会論文誌，Vol.39, No.7 (1998)

[10] 宮守:「アメリカ契約法入門」，中央経済社(1998)

[11] Santos and Carmo: "A Deontic Logic Representation of Contractual Obligations", in John-Jules CH. Meyer and Roel J. Wieringa(Eds.): "Deontic Logic in Computer Science: Normative System Specification", 1993.

[12] 飯島 正，根本 知幸，山吉 育子，浦 昭二:「開発方法論 SSADM のオブジェクト指向モデル化と その利用の構想」，情報処理学会，情報システム研究会(1992 年 9 月)．

[13] 飯島，山本，土居:「共生・寄生エージェントモデル(S/PAM)に基づくロボットプログラミング」，電子情報通信学会知能ソフトウェア工学研究会(1999 年 9 月)

[14] Hideyuki Nakashima,and Itsuki Noda: Dynamic Subsumption Architecture for Programming Intelligent Agents, ICMAS98(1998).

[15] Bertrand Meyer:オブジェクト指向入門，アスキー出版局(1990)

[16] W. ブリー:デザインパターンプログラミング(増補ッパン(1998)