電子商取引プロセスにおける 電子契約実行支援のためのメッセージ交換モデル

蒋 海鷹† 岩井原 瑞穂‡ 上林 弥彦‡

†‡京都大学大学院情報学研究科 〒606-8501 京都市左京区吉田本町

E-mail: †jiang@db.soc.i.kyoto-u.ac.jp, ‡{iwaihara, yahiko}@i.kyoto-u.ac.jp

あらまし 現在，電子ビジネスのための XML 言語の標準化として ebXML などが提案されており，複数の企業のワークフローを接続するメッセージ仕様が検討されている．電子契約に基づいて実行されるワークフローに例外が発生した場合は，電子契約にある規約に基づいて解決されなければならないが，これはアドホックなプロセスであり，人間の対話による解決が主体となる．これに対し電子契約に基づいたメッセージ交換のテンプレートを用意することにより，問題解決の支援を行なうことが考えられる．本稿では，そのような電子契約プロセスと対話プロセスのモデルを提案する．

キーワード 電子商取引，ワークフロー，協調作業支援，電子契約

Message Exchange Model for Supporting Electronic Contract Execution in E-business Processes

Haiying JIANG† Mizuho IWAIHARA‡ Yahiko KAMBAYASHI‡

†‡Graduate School of Informatics Kyoto University Yoshida-Honmachi, Sakyo-ku, Kyoto, 606-8501 Japan

E-mail: †jiang@db.soc.i.kyoto-u.c.jp, ‡{iwaihara, yahiko}@i.kyoto-u.ac.jp

Abstract The standardizations of XML language for e-business like ebXML are being carried out and message specifications for connecting workflows of several companies are also being examined. Exceptions should be resolved by complying with the rules of an electronic contract when they occur in those workflows. This is an ad hoc process and requires human communication and decision. In this paper, we propose a model which integrates workflows, e-contracts and solution processes to resolve exceptions by following e-contracts and making decisions through communication templates.

Keyword e-business, workflow, electronic contracts, e-commerce, CSCW

1. はじめに

現在，ウェブサービス技術の標準化が行われている．その中で，SOAP[17]，WSDL[20]，UDDI[18]の三つの標準を組み合わせることによって，ウェブサービスを展開することが，有力な標準案となっている．WSDL でのプロセス表現力の不足を補うために，WSDL を拡張する BPEL4WS[2]の仕様も公開されている．BPEL4WS の仕様書ではプロセスの表現力を提供する同時に，ウェブサービスが実際に利用される段階で，サービスの実行を監視すること，そして例外処理の重要性も指摘している．特に，B2B 分野では，企業間の電子契約の実行は，注文書，請求書などの電子文書を含んだメッセージの交換に基づいている．そこでは，メッセージ転送中のエラーなどの通信レベルでの例外もあるし，複数の参加者の間で電子契約が締結された時に予想されなかった例外が発生する可能性もある．その場合，再送信や，通信エラー情報の送信といった通信レベルのプロトコルは問題解決には用いることはできず，人間の判断に基づく例外処理が必要になる．

このような例外処理を支援するために，本稿では電子契約プロセスと対話プロセスのテンプレートを用いた，Workflow-Contract-Solution(WCS)モデルを提案する．以下，2 節で現在提案されている電子商取引用 XML 言語についてまとめ，3 節と 4 節で，本稿で提案する WCS モデルと問題解決プロセスを述べる．5 節は関連研究であり，6 節はまとめである．

2. 電子商取引用 XML 言語

ウェブサービスを B2B 分野に拡張させるために，ebXML[9]の技術仕様では B2B のアーキテクチャの構築方法が紹介されている．

電子商取引プロセスを実行するワークフローは企業内部のワークフローと企業間のワークフローがある．ebXML のビジネスプロセス仕様[10]を利用して，設計するのは，主に RosettaNet[16]での PIPs のような企業間のワークフローであると考えられる．設計されたビジネスプロセスは，CPP(Collaboration-Protocol Profile)に参照される．各プロセスがどの通信プロトコルで展

開されるか，ということを決めるのは CPP の役割である．二つの CPP から CPA(Collaboration-Protocol Agreement)が合意される．契約が抽象性の高いため，あらかじめ設計されたビジネスプロセスを参照する CPP/CPA により，アドホックな例外問題を解決するのは，困難である．そこで，電子契約は一部の設計された企業間ワークフローに参照される ebXML の CPP/CPA モデル[11]により表現できず，CPP/CPA と関係があるが，独立のものであると考えられる．合意された CPA に基づいて，企業間のワークフローは BSI(Business Service Interface)アプリケーションにより展開される．その CPP/CPA は BSI アプリケーションの設計ファイルであると考えられる．

外部から見れば，BSI は電子商取引プロセスの実行を担当する部分であり，実行時に CPA に参照され，一つのビジネス・アクティビティで一つの役割(Role)を果たす．各取引相手は，決められた役割を参照しながら，その BSI を通じて，ビジネスサービスを提供したり，ビジネスサービスを受けたりして，電子商取引プロセスを展開する．実行段階で，その BSI がビジネス・ルールに基づいて，時間超過や例外などによる失敗を発見するなどの機能を持つべきであると ebXML は想定しているが，具体的に BSI がどのように働くかについては定義されていない[10].

つまり，企業間のワークフローの例外を検知し，契約に基づいて，ワークフロープロセス(内部ワークフロー，外部ワークフロー，または両方)の調整などの方法により，例外処理をするのは，BSI の役割である．そのため，BSI は企業間のワークフローと企業内部のワークフローとの接続口であり，ある意味でワークフロー管理システムの一種でもあるといえる．通信レベルの例外を発見するだけではなく，アドホックな例外を BSI に処理させるため，通信レベルでの例外しか発見できない現在の BSI モデルを拡張する必要がある．拡張される BSI は電子契約と意味的なリンクを設けることができ，それによって契約の実行を監視して，何かアドホックな例外が発生した場合，契約に基づいて，対話プロセスにより状況を分析し，そして解決方法を設計し実行する機能を持たせることが考えられる．

以下，BSI モデルの例外発見と例外処理機能を拡張する形での，対話による問題解決支援を特徴とした，電子契約処理モデルを検討する．

3. Workflow-Contract-Solution モデルによる電子契約処理

電子契約に基づいてビジネスを行なう組織の集合(例えば 2 つの企業)について，組織内でのワークフローと組織間のワークフローを区別することができる．この場合，組織間のワークフローは電子契約の実行を担う部分であるといえる．

3.1. 電子契約処理の特徴

(論理性) 電子契約の規則は，論理的かつあいまい性の少ない表現が必要である．論理的な構造は意味が明確な言語により表現することが必要である．

(高い抽象性) 電子契約の規則は抽象性の高い記述が含まれ，契約中のある規則を適用するためには，実行時に得られる多くの情報を必要とする．

(人間による判断) 規則の適用には，規則の前提条件に該当するかの判断や，規則が適用されるかの解釈に人間の判断を必要としている．また規則が存在しない場合も協議が必要となる．

(対話による問題解決) 例外処理など，問題解決を行なう際に，参加者同士による協議が行なわれ，解決方法を決定し，合意事項を記録する必要がある．この対話のプロセスをシステムで支援する必要がある．

3.2. Workflow-Contract-Solution モデルによる電子契約処理の概要

電子契約の実行支援を行なうための Workflow-Contract-Solution モデル(WCS モデル)について，まず概要を述べる．WCS モデルは以下の 3 つの要素および各要素間の関連からなる．

Workflow (ワークフロープロセス) ワークフローは契約を実行するために，組織内部および組織間での作業手順を示すプロセスである．通常はワークフローに定義された作業を行なえば，契約を完了することができる．しかしワークフローに想定していない状況が発生した場合は，例外ととらえられる．あらかじめ全ての例外を想定することは困難であるため，例外ごとにアドホックな問題解決を行なうことが必要となる．

Contract (契約プロセス) 電子契約の内容を表現するプロセスである．行為や義務や権利などの要素，およびこれらの要素の依存関係を表すリンクで表現される．契約の参加者であらかじめ合意された契約プロセスのテンプレートを用意しておき，契約を摘要するたびに契約プロセスのインスタンスを生成する．契約プロセスはワークフロープロセスと異なり，具体的な作業手順を記述するものではなく，義務や権利を抽象的に表現するものである．例外が発生した場合に，それが契約プロセスに記述のあるどの状況かを当事者の議論によって定め，その状況に対応した解決策を契約プロセスから求める．

Solution (問題解決プロセス) 契約実行における参加者間の交渉や議論の実行を行なうプロセスである．例外状況が発生したとき，その解決方法を当事者間で議論し，解決策について意志決定を行なう．その際，契約プロセスを参照し，可能な解決策を選択する．これにより契約に従った問題解決を支援する．解決策は

新たなワークフローの定義，あるいは既存のワークフローの一時的または恒久的な変更として表現される．

WCS モデルは，上記の 3 つのプロセスで役割を分担しながら電子契約の実行を行ってゆくものである．以下，各構成要素の説明を行なう．

3.3. ワークフロープロセスと抽象プロセス

プロセスとは，ある目的のために構成された一連の作業(アクティビティ)の手順である．**アクティビティ**は，人間によるアクティビティとプログラムにより自動実行されるアクティビティが混在することができる．アクティビティはまた 1 つのプロセスであってもよく，入れ子構造のプロセスを形成することができる．

プロセステンプレートとは，あるプロセスの集合を特徴づける情報であり，あるテンプレートに属するプロセスをそのテンプレートのインスタンスであるという．例えば，プロセスの参加者や，作業内容は変数としてテンプレートに表現しておき，実行時にこれらの情報をテンプレートに代入することにより，プロセスのインスタンスが生成される．

プロセスの構成要素として，アクティビティ，およびアクティビティ間の制御ノードとして逐次実行(sequence)，並列分岐ノード(AND-split, OR-split)，合流ノード(AND-join, OR-join)，条件分岐ノード(if-then-else)がある．プロセスの形式的モデルの詳細な議論は本稿では行なわない．

本稿では，ワークフロープロセスと抽象プロセスというプロセスの分類を用いる．ワークフロープロセスは具体的な作業手順を表現するのに対し，抽象プロセスは，電子契約など具体的作業をより抽象化したプロセスを指すものとする．また抽象プロセスは後述の意味的リンクを持つことができる．

プロセスの実行順序を表現するリンクの他に，プロセス間の意味的な関係を表現する**意味的リンク(semantic link)**を用いる．意味的リンクには意味を表す型名を持たせる．実行順序のリンクは，逐次実行や並列実行といった制御構造を与えるが，意味的リンクはプロセス間の制御構造以外の依存関係や対応関係を表現する．意味的リンクの例として，後述する型名<executes>は，C <executes> D により，ワークフロープロセス C は，電子契約中に定められたプロセス D を実行するものであるなどがある．意味的リンクにさらに特殊な実行制御の意味を与えることがある．型名を持たないリンクは，意味的リンクではなく逐次実行を表わすものとする．

電子契約の表現では，状況や因果関係，権利や義務など，ワークフローによる作業の表現よりも抽象的な記述が含まれる．電子契約の表現のために**抽象プロセス(abstract process)**を用いる．抽象プロセスとは，通常のプロセスと同様の構成要素である子プロセスの集合と制御ノードに加え，意味的リンクによる子プロセス間の関連集合を持つ．さらに，当該抽象プロセスから外部の抽象プロセスやワークフロープロセスへの意味的リンクも持つことができる．ワークフロープロセスは，外部とインターフェースとなるノードを持ち，整然とした入れ子構造を持つのに対し，抽象プロセスでは他の抽象プロセスやワークフロープロセスを意味的リンクにより自由に参照できる有向グラフ構造を持つものとする．これによりワークフロープロセスのような具体的作業と，契約上の抽象プロセスを意味的リンクで対応付けることを行なう．

ワークフローの実行状況が電子契約に定められた条件が成立することなどの表現に抽象プロセスを用いる． また，プロセス間の優先順位や排他条件など，契約条項間の論理条件の表現に用いる．それらの例として，以下の<excludes>と<alternative>の抽象プロセスの例を上げる．

<excludes> これは契約の適用が除外されるなどの条件を別のテンプレートで記述するための関連である．A <excludes> B により，A の条件が成立しているとき，B は適用されない．

<alternative> A <alternative> B (代替) プロセス A が「有効」でないとき，プロセス B が有効になる．

3.4. 契約プロセス

電子契約は契約プロセスの集合であり，契約プロセスは抽象プロセスの一種とする．契約プロセスにより，電子契約を構成する各規則を記述する．電子契約は，契約実行前にあらかじめ定義されるものとする．契約プロセスのテンプレート集合として電子契約を表現し，電子契約による処理を行なうときは，然るべきテンプレートから契約プロセスのインスタンスを生成する．

契約プロセスの例を文章で表現したものを下記に示す．また，そのうちの一つを図式化したものを図 1 に示す．

① 売主が納品期限に間に合わない場合，買主は注文をキャンセルすることができる．さらに，違約金を売主に要求することができる．
② 売主が納品期限に間に合わない場合，間に合わないことが分った時点で買主に通知すべきである．
③ 商品の輸送で不良品が出る場合，売主は証拠を買主に提示すべきである．
④ 買主は商品を確認した後，商品に対する責任は買主に移る．
⑤ 買主は商品を確認するときに，商品確認基準が契約での基準より厳しいと，売主は判断したときに，買主に異議をすべきである．

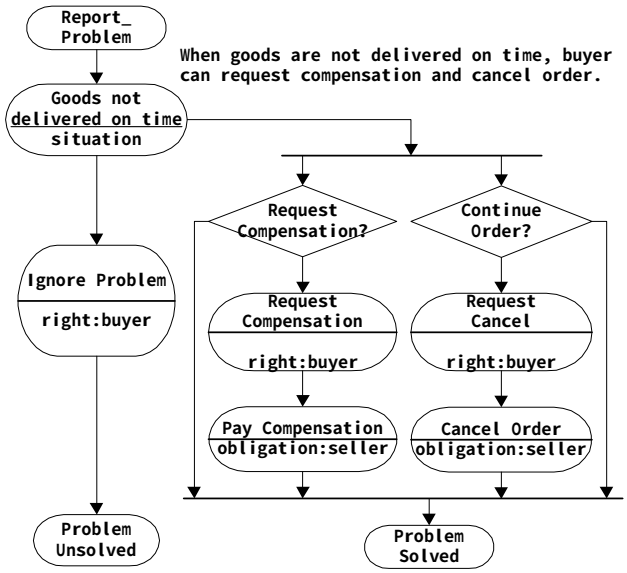

図 1　契約プロセステンプレート

契約プロセスのテンプレートでは，各プロセスにその**実行者(actor)**を表わす属性を与える.典型的なのは，買い手および売り手というパラメータが用いられるであろう.

契約プロセスを構成する要素プロセスは，その機能や目的に応じて以下の種類に分類する．以下では特に断わらない限り，プロセスとは抽象プロセスのことを指す.

[状況(situation)プロセス] 契約において,ある特定の条件が成立したときにある処理を行なうなどの規則において，契約の実行中に特定の状況が成立したこと表現するプロセスである．特定の状況として，例外やイベントが発生を用いることができる.

[義務(obligation)プロセス] プロセス実行者が行なうべき義務の内容を表わすプロセスである.

[権利(right)プロセス] プロセス実行者が権利として与えられる行為を表わすプロセスである.

[禁止(prohibition)プロセス] プロセス実行者が行なってはならない行為を表わすプロセスである.

上記以外のプロセスとして，開始点や終了点，条件分岐などのプロセスがあり，これらは通常プロセスとする.

テンプレートを表す図では，状況，義務，権利，禁止を機能分類という属性の値として記入する.

一般に契約プロセスは，複数の規則の集合からなり,規則の間の優先順位も規定されている場合がある．これはあらかじめ合意された電子契約の一部として，プロセステンプレートとして個々の契約プロセス間の優先順位を抽象プロセスで記述することができる．例えば<alternative>の抽象プロセスを用いて,ある契約プロセス A の適用が優先的に検討され，次に契約プロセス B の適用が検討される．また，契約プロセスに定義された例外処理の階層の表現に用いる.

3.5. 対話プロセス

対話プロセス(communicative process)は，電子契約の実行において，参加者間の交渉や議論などを実行する抽象プロセスである.

ワークフローの不得手である柔軟かつアドホックな参加者間の交渉や議論などのプロセスである.

対話プロセスでは，参加者が発言することがプロセスにおける主要な行動であり,その発言は performative とも呼ばれる.対話プロセスでは個々の発言(メッセージ)の発言目的を表わすプロセステンプレートを用いる．テンプレートの例として, propose(提案), claim(主張)，question(質問)，request(要求)，challenge(反論), acknowledge(通知)等がある．対話プロセスは構造化議論モデル(IBIS)[3]およびメッセージトランザクションモデル[13,14]を参考にすることができる.

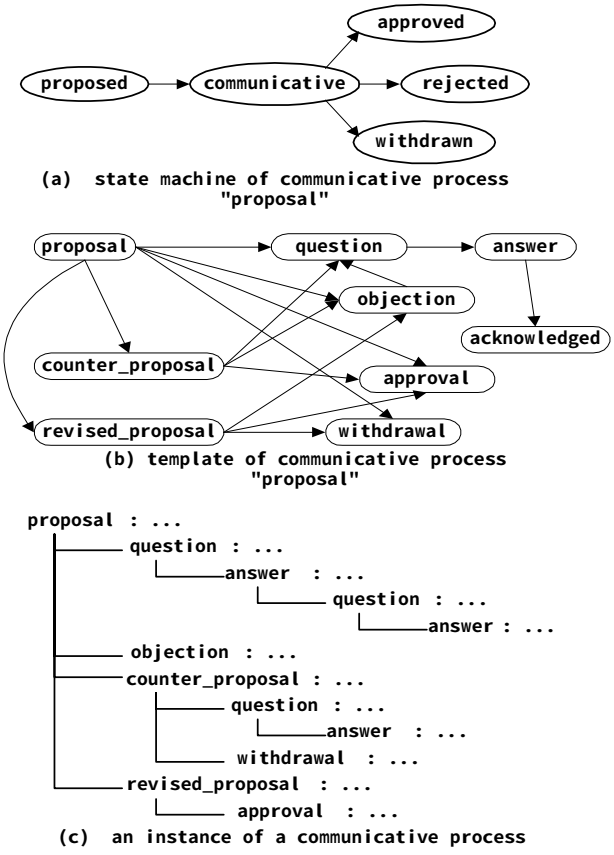

図 2　対話プロセス

対話プロセスには，1 つの状態機械を割り当てる．その状態は開始，議論中およびいくつかの終了状態とし，対話プロセスの進行状況を表わす．この状態機械により，議論の進行状況を表現する．図 2(a)は対話プロセス"propose"の状態機械である．対話の進行には，対話プロセスのテンプレートを用いる．一般に人間同士の対話は，自由に発展してゆくものであり，ワークフロープロセスのように制御構造を事前に固定するのはなじまない．対話の意図を明確にするために，個々

の発話に performative というタグ名を発話者につけさせるようにする．そして，各発話が question-answer のように対応が取れるように，ゆるやかな構造化を導入するために，図 2(b)のようなテンプレートを用意する．このテンプレートは，ある performative タグの次に来ることのできる performative タグを規定する有向グラフである．発話者は，このテンプレートに従って，ある発話に自分の発話を続ける形で議論を進行させる．図 2(c)の例に示すような，掲示板のような木構造として，議論が展開されてゆく．議論の終了を表わすタグ(図 2(b)では approval や withdrawal)が発話されたら，前述の状態機械の状態遷移を起すようにする．発言権や決定権といった制御方法は，このテンプレートに付加する．意志決定方法の違いに応じて異なる対話テンプレートを用意する．

対話プロセスとワークフロープロセスや契約プロセスの間では，以下の意味的リンクを用いる．

<argues> (逆方向: <argued_by>)　議論を行なう対話プロセスの発話から，議論の対象となっているプロセスへのリンクである．まだ議論途中で結論となっていない対話プロセスの要素がリンク元となり得る．

<approves> (逆方向: <approved_by>)　議論プロセスにおいて，「合意」ノードなど，結論が得られたノードから，その合意事項である対象へのリンクである．

4. 問題解決プロセス

電子ビジネスの実行中に生じた問題が，ワークフローでは対処方法が未定義の例外として検出された場合，何らかの解決を行なわなければならないが，その際，電子契約に基づいた解決方法にとることが契約遵守の上で必要である．電子契約に基づいて解決するプロセスを問題解決プロセスとし，そのモデルを議論する．問題解決プロセスを実行することにより，例外で発生した問題の処理を行なう．問題解決プロセスは，発生した問題ごとにインスタンスが生成される．

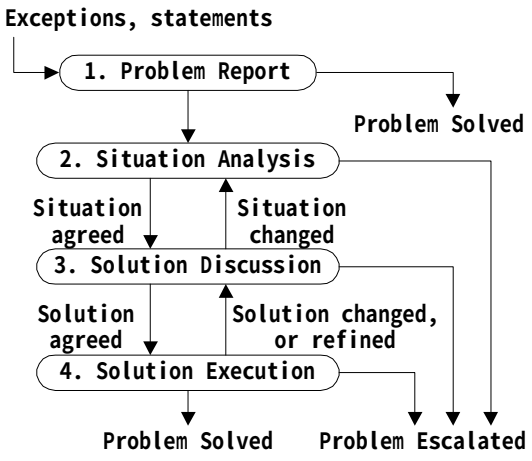

図 3　Problem solution process

図 3 に示すように，問題解決プロセスを，(1) 問題提起ステージ，(2) 状況分析ステージ，(3) 解決策検討ステージ，(4) 解決策実施ステージの 4 つの段階的なステージで表現する．図 4(a)は，状況分析ステージ，(b)は解決策検討ステージを表す．以下，各ステージを説明する．

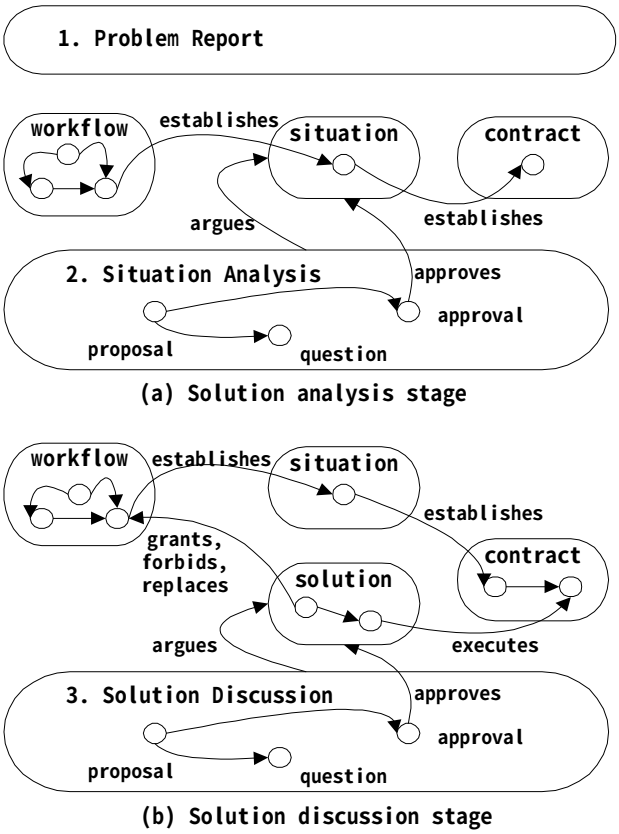

図 4　状況分析ステージと解決策検討ステージ

4.1. 問題提起ステージ

ワークフローに定義のある例外が発生した場合や，期待していた相手側の行動が行なわれないなどの場合に，まず関係者に問題を通知し，確認を求める対話プロセスが問題提起(problem report)ステージである．

問題提起ステージが開始される要因は，時間切れなどワークフローでのエラーとしてシステムが提起するものと，対話プロセスにおいてある参加者の発言が，将来的に他方の参加者に不利益を与えることが予想されるため，相手に説明を求めるなど，発言に起因した問題提起が考えられる．前者は，ワークフロー中に例外が発生したら問題提起ステージを開始するようにあらかじめ定義しておくことが考えられる．

問題提起の段階で，相手が対策を講じることにより問題が解決されれば，これで問題解決プロセスは終了する．もしこのように単純に問題解決できない場合は，対策レベルをエスカレートさせ，次の状況分析ステージを当事者が開始することになる．

4.2. 状況分析ステージ

状況分析(situation analysis)ステージは対話プロセスである．まず提起された問題に至った状況をワークフローの実行状態や過去に発信されたメッセージをevidence として提起者が提示する．そしてその状況を電子契約にある状況プロセスに後述の意味的リンク<establishes>で対応づける．また，問題の別の当事者(concerned parties)が異なった状況解釈を提示することもある．状況に関する認識の違いは，当事者同士の対話により明確化される．状況分析について，当事者の共通の認識が得られれば，解決策検討ステージに進むことができる．もし，共通認識に至らなければ，他の紛争処理制度や，司法制度などの本システム外の問題解決手段に移行することになる．

4.3. 解決策検討ステージ

解決策検討(solution discussion)ステージは対話プロセスとして実行される．図 4(b)が示したように，提起された問題を解決する方法(解決策)を，関係者が抽象プロセスである解決策プロセス(solution process)として記述し，提案する．つまりここでの対話プロセスはプロセスを設計するためのプロセスである．解決策プロセスの記述とは，抽象プロセスを記述し，意味的リンクをワークフロープロセスとの間に設定することである．提案の修正を経て，合意が得られれば，解決策として承認されたことになり，解決策実施ステージに移行する.提案された解決策で合意が得られなければ，本システム外の問題解決手段へ移行する．

検討の途中で新たに分った事実が提出されたり，状況の解釈が修正・追加されたりするなど，必要に応じて解決策検討ステージから状況分析ステージへ戻り，反復しながら実行される．

契約プロセスは一定の条件のもとで当事者の権利や義務を定める抽象性の高い記述である．検討中の問題において，解決するために必要な権利の行使や義務の履行を実現する解決策プロセスを，解決策検討ステージで合意することになる．

解決策プロセスは，具体的な行動を定義するワークフロープロセスとする方法がある一方で，大まかな解決方針を抽象プロセスとして表現しておいてその具体化を解決策実施ステージで行なう方法もある.後者は，大まかな解決方針を解決策検討ステージとして合意しておき，実際の解決手段は後にあるいは実行時に動的に決めてゆくものである．後者の解決策プロセスを抽象解決策プロセスと呼ぶことにする．抽象解決策プロセスは，具体的なワークフロープロセスに結びつけられる必要があるが，その場合は対話プロセスで合意を求める．

4.4. 解決策実施ステージ

契約と合意に基づいた解決策プロセスを実行するのが解決策実施ステージ(solution execution)ステージである．解決策プロセスの実行中に，解決策の不具合が判明したときは，前段階の解決策検討ステージに戻り，解決策の修正を検討する．また前節の抽象解決策プロセスを解決策実施中に具体化する場合も，解決策検討ステージに戻り検討する．

4.5. 状況プロセス

状況プロセスについてさらに検討する．状況プロセスはワークフロープロセスの実行状況，および対話プロセスの内容について解釈を与え，契約に記述のある状況と対応づけを表現する抽象プロセスである．状況プロセスは，作業の手順を表現するものではなく，ある状況が成立していることを表現するため のものである．そして，ワークフロープロセスおよび対話プロセスの構成要素から抽象プロセスへの型名<establishes>のリンク，および条件ノードから契約プロセスの状況プロセスへ型名<establishes>のリンクも状況プロセスに含まれる．<establishes>の逆方向の型名として<established_by>を用いる．条件ノードを経ずワークフロープロセスや対話プロセスの要素から直接契約プロセスの状況プロセスを参照してもよい．すでに起きた状況を記述するものとするため，リンク<establishes>の元は，実行済みのワークフロープロセス(の要素)または対話プロセスの行なわれた発言でなければならない．リンク<establishes>は，契約中のある状況プロセスとそれを成立させているイベントを対応づけるものである．

状況プロセスのライフサイクルとして，状況プロセスのインスタンスは図 5 の状態遷移に従うものとする．状況プロセスの状態名という属性として，下記の状態名を持たせる．

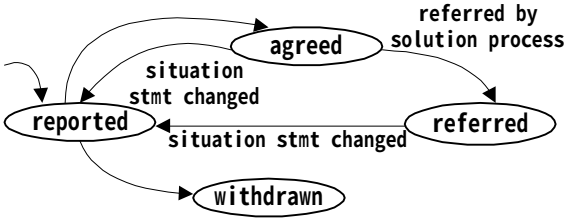

図 5　State machine of a situation process

提起状態(reported): ある状況プロセスが問題提起された．当事者の間で承認は得られていない状態である．

合意状態(agreed): 対話プロセスにより当事者による合意が得られ，契約上のある状況が成立していると認定された状態である．新事実の発見などで状況の解釈が変わった場合は，状況プロセスの修正となり，その場合は提起状態に戻る．

参照状態(referred): 合意の得られた状況プロセスが，それを基にある解決策プロセスから参照されてい

る状態である．この状態で状況プロセスを修正する必要が生じた場合は，解決策プロセスを一旦 abort しなければならない．修正した場合は，提起状態に戻る．

取り下げ状態(withdrawn): 状況プロセスが取り下げられた．

4.6. 解決策プロセス

解決策プロセスは，解決策の実行手順を表現するプロセスである．解決策プロセスとワークフロープロセスおよび契約プロセスとの間に以下の意味的リンクの集合を設ける．

契約プロセスに定められた権利や義務を，解決策プロセスのそれを実行する要素プロセスにリンク型<executed_by>のリンクおよびその逆方向のリンク名<executes>で対応付ける．A <executed_by> B により契約プロセスの義務あるいは権利を表わす要素 A が，解決策プロセスの要素 B で履行されることを明示する．逆方向の<executes>は B を実行する根拠が A であること解釈する．後の議論において，<executes>のリンクに，実行制御のセマンティクスを与える．

一般に契約プロセスの間には何らかの優先順位が事前にテンプレートで設定されていると考えられる．一方，契約プロセスには抽象的な状況の記述が含まれるものであり，複数の規則が該当しどの規則が適用されるべきかは，あらかじめ明示されていないことがある．この場合，どの規則により問題を解決するかを議論し，個々の解決策では，どの規則を適用するかを表現する必要がある．このため，解決策プロセスには，その解決策プロセスにおいて契約プロセスの優先順位についても抽象プロセスで表現する．

解決策プロセスの記述では，実行途中のワークフロープロセスあるいは解決策として新たに導入されたワークフロープロセスに対し，修正や置き換え，あるいは実行制御の変更を行なう．そのために図 4(b)に示したように下記の意味的リンクを用いる．

<grants> 解決策プロセスのある要素から，あるワークフロープロセスの実行を許可する．リンク元のプロセスが実行されたとき，リンク先のプロセスが実行可能になるという実行制御のセマンティクスを持つ．リンク元が実行されるまでリンク先の実行を停止する働きを持つ．

<forbids> 解決策プロセスのある要素が，ワークフローのあるプロセスの実行を禁止する．リンク元のプロセスが実行されると，リンク先のワークフローのプロセスは実行できなくなるという実行制御のセマンティクスを持つ．禁止の根拠を求める場合は，解決策プロセスから契約プロセスへの<executes>リンクで求めればよい．

<replaces> ワークフロープロセスの部分を，解決策プロセス内に定義されたワークフロープロセスの断片で置き換える．通常のワークフロープロセスの一時的変更として捉えられる(参照).

<grants>と<forbids>のリンク先については，抽象プロセスは用いず，ワークフロープロセスのみとする．抽象プロセスにより複数のプロセスのいずれかが<grants>により実行可能になる，といった disjunctive な実行制御は行なわない．ワークフロープロセスのある集合すべてに<grants>を与えるには，それらのプロセスすべてへの<grants>のリンクを張ればよい．

解決策プロセスのライフサイクルとして，合意が得られたものであるかの有無や，実行が成功して終了したかを識別するために, 以下の状態機械を導入する(図 6).

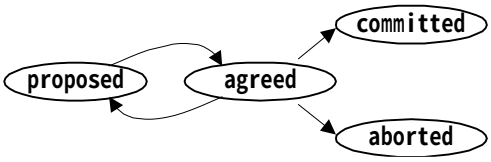

図 6　State machine of a solution process

提案状態(proposed)：提案として解決策プロセスが生成されたときの状態であり，参加者の承認は得られていない．

合意状態(agreed)：対話プロセスにより，合意が得られ，解決策プロセスとして承認されたもの，解決策プロセスの修正や抽象解決策プロセスの詳細化を議論する対話プロセスが開始された場合は，提案状態に戻る．

コミット状態(committed)：解決策プロセスが成功して終了した状態である．

アボート状態(aborted)：解決策プロセスの実行が成功せずに終了した状態である．

5. 関連研究

WCS モデルは複数ワークフローの連携, 電子契約モデル，例外処理，対話システムなどの研究分野に関連している．

複数のワークフローを連携する研究については，CrossFlow モデル[12]と ICP モデル[7]がある．これらの研究の共通点としては，契約をベースにして，各側のワークフローを Role で区別し，実行時にその Role にマッピングしながら両側のワークフローを連携するという標準化活動 RosettaNet モデルに近いアプローチを取っている．しかし，それらのモデルでは，契約実行中に例外が発生した場合の解決方法という課題は議論されていない．

契約を表現するいわゆる契約モデルに対する研究については，CrossFlow の研究と関連して，Who，What，Where，How などの要素を定義するモデル[1]があり，時間順序を用いて義務の変換を表現する電子契

約モデル[15]もある．それらのモデルは例外処理の方式を取らず，契約実行時の問題解決方法を議論していない．それに対して，われわれの WCS モデルは当事者が電子契約を参照することにより問題解決方法を決め，電子契約，ワークフロー，解決策プロセスをリンクで接続して意味づけを表現する方法を取る．これは CISG[8]など実際の契約にも適用可能である．CISG が通常の契約のように抽象性の高い表現が多く，普通のワークフローにより CISG を表現するには限界があるため，抽象プロセステンプレートで設計するのがわれわれのアプローチである．

例外処理に関する研究は広く行なわれている．われわれの注目するのは予想されない例外に対する協調例外処理である．再発する可能性のある予想されない例外に対しては，ワークフロー定義を調整する必要があるという基準[4]を採用する ADOME-WFMS モデル[6]がある．そのモデルは以前に発生した例外が今回の例外のシナリオに近い場合，その時の例外処理を代替案としてユーザに提示して，ある程度の協調例外処理を実現する方法を提案している．しかし，われわれの WCS モデルで扱う例外は，人間の判断が必要な例外であるため，以上の例外処理方法は，本研究の目指すアドホックの例外を処理するのに，不十分である．

対話を利用したワークフロー管理システムの拡張モデル[19]と対話による契約実行時のイベント処理モデル[5]が提案されている．ただし，これらのモデルは，契約に基づく問題解決とワークフローへのリンク付けなどの問題解決ための方法を説明していない．われわれの提案した WCS モデルは以上の未解決の課題について，必要な要素を組み合わせたより適応能力の高い電子契約実行支援モデルとなっている．

6. まとめ

本稿では，電子商取引を行なうワークフローにおいて，予想されない例外などワークフローに記述のない問題が発生した場合に，事前に合意した電子契約のプロセステンプレートに従って，対話プロセスにより問題解決プロセスを議論し，問題解決プロセスを段階的に詳細化する WCS モデルを提案した．今後は，プロトタイプによりモデルの効率と実用性を検証する．

文　献

[1] S. Angelov, P. Grefen; “A Framework for the Analysis of B2B Electronic Contracting Support;” Proceedings 4th Edispuut Conference, Multidisciplinary Perspectives on Electronic Commerce; Amsterdam, Netherlands, 2001.

[2] Business Process Execution Language for Web Services 1.0. July 2002. www-106.ibm.com/developerworks/webservices/library/ws-bpel/

[3] Conklin, J. and Begemau, M. L., "gIBIS: A Hypertext tool for Exploratory Policy Discussion, " ACM Trans. Office Information Systems, Vol. 6, No. 4, pp. 303-331, pp. 1-11, 1988.

[4] F. Casati, “A discussion on approaches to handling exceptions in workflows,” CSCW Workshop on Adaptive Workflow Systems, Seattle, WA, USA, 1998.

[5] Martin W.A. Caminada, “Towards a formal model for contract execution,” Proceedings of the 4th International Workshop: The Language Action Perspective on Communication Modeling, Copenhagen, Denmark, September 12-13, 1999.

[6] Dickson K. W. Chiu, Qing Li, and Kamalakar Karlapalem, “Web Interface-Driven Cooperative Exeption Handling in ADOME Workflow Management System,” WISE 2000, pp174-182, Hong Kong, China, June 19-21, 2000.

[7] Qiming Chen, Umeshwar Dayal, and Meichun Hsu, “Conceptual Modeling for Collaborative E-business Processes,” 20th International Conference on Conceptual Modeling, Yokohama, Japan, November 27-30, 2001.

[8] CISG 日本語版. www.cisg.law.pace.edu/cisg/text/japanesetext.html.

[9] Electronic Business XML. www.ebxml.org/

[10] ebXML Business Process Specification Schema 1.01. May 2001. www.ebxml.org/specs/ebBPSS.pdf

[11] ebXML Collaboration-Protocol Profile and Agreement Specification 1.0. May 2001. www.ebxml.org/specs/ebCCP.pdf

[12] P. Grefen, K. Aberer, Y. Hoffner, and H. Ludwig; “ CrossFlow: Cross-Organizational Workflow Management in Dynamic Virtual Enterprises;” International Journal of Computer Systems Science & Engineering, Vol. 15, No. 5, pp. 277-290, 2000.

[13] 井上創造, 岩井原瑞穂, “非同期型コミュニケーションにおけるトランザクションの動的構築”, 情報処理学会論文誌: データベース, Vol.˜ 40, No.˜ SIG 8 (TOD 4), pp. 1-12, 1999.

[14] 井上創造, 岩井原瑞穂, “コミュニケーションを用いた作業プロセスの動的な獲得”, 情報処理学会論文誌: データベース, Vol.˜ 42, No.˜ SIG 15 (TOD 12), pp. 50-62, 2002.

[15] O. Marjanovic, Z. Milosevic, “Towards Formal Modeling of e-Contracts,” 5th IEEE International Enterprise Distributed Object Computing Conference (EDOC 2001), Seattle, Washington, USA, September 04-07, 2001.

[16] RosettaNet. www.rosettanet.org/

[17] Simple Object Access Protocol (SOAP) 1.1. May 2000. www.w3.org/TR/SOAP/

[18] UDDI: Universal Description, Discovery and Integration. www.uddi.org/

[19] H. Wedekind, “Extending workflow management systems by a dialogical component,” Proceedings of the 1996 ACM symposium on Applied Computing, p.150-157, Philadelphia, Pennsylvania, USA, February 17-19, 1996.

[20] Web Service Description Language (WSDL) 1.1. May 2001. www.w3.org/TR/wsdl