オブジェクト指向ペトリネットによる ワークフロー管理システムの構築

Building a Workflow Management System based on Object-Oriented Petri Net

片山輝彦† 金子良太‡ 飯島正‡
Teruhiko Katayama† Ryota Kaneko‡ Tadashi Iijima‡
†慶應義塾大学 大学院 理工学研究科
‡慶應義塾大学　理工学部
† Department of Social Informatics, Aoyama Gakuin Univ.
‡ Faculty of Information and Communication, Bunkyo Univ.

要旨

ここ数年にわたって構築中の，オブジェクト指向ペトリネットに基づくワークフロー管理システムの基礎と，それに関連する研究プロジェクトについて紹介する．ここで使用するオブジェクト指向ペトリネットは，Nets-within-Nets 意味論に基づく Reference Net をベースとして，独自に拡張を重ねているものである．このプロジェクトは現在進行中であり，今回は，分散開発を意図し共同執筆機能を備えたエディタ，サーバとしてワークフローの実行を管理するワークフローエンジン，ワークフロー参加者が携帯できる Android 端末で実行可能なワークフローモニタを中心に概観する．

1. はじめに

ワークフローとは業務の流れであり，複数の人間やコンピュータ（サーバ）等のリソースによって実行される作業（タスクないしアクティビティ）からなるプロセスである．一般に，業務の流れには人間や機械，書類，生産物など様々なリソース（実行のための資源，情報を提供する資源など複数に分類できる）が複雑に絡み合っている．そのためワークフローは，それらリソースや作業間の関係，必要な制約条件などがわかりやすく整理され表現されなければならない．

これまで，ワークフローをモデル化するための研究は，数多く行われてきた．その中の一つにペトリネットによるモデル化がある（たとえば YAWL[1]）．ペトリネットは数学的に厳密に定式化されているため，到達可能性や有界性といった各種の性質を検証できるという利点を持ち，意図通りのワークフローが表現できているか，欠陥が残っていないかを確認する上で有効である．一方で BPMN[2]といった曖昧性を許容する表現に比べると直観的な理解容易性を損ない，ネットワークのサイズも大きく複雑になることも多い．しかし，人間の実施する作業だけでなく，コンピュータが実施する各種サービスとの協調連携を考えた自動化（いわゆる　ワークフローオートメーション）を想定したワークフローのモデリングにあたっては，要求獲得から上流モデリングの過程であればともかく，少なくとも最終的な実行レベルでは厳密性を導入せざるをえない．さらには多数のサーバ，多数の参加者を協調連携させるようなワークフローのためには，多重レベルに渡る例外フローの導入とそれを下支えする検証能力は不可欠であり，大規模化と複雑化は避けることができない．

そこで，本研究ではまず，ワークフローを表現するにあたり，基本的にオブジェクト指向概念に基づくモジュール性を取り入れたオブジェクト指向ペトリネットを用いる．モジュール性を取り入れることで，大規模化と複雑化を軽減することを意図している．オブジェクト指向ペトリネット(オブジェクトペトリネットとも呼ぶ)には，いろいろなタイプが存在するが，本研究プロジェクトと複数の関連プロジェクトでは， nets-within-nets 意味論[3]にもとづく Reference Net をベースとして，各種の応用に向けて数年にわたって独自の拡張を重ねてきたオブジェクト指向ペトリネット仕様 OPeN (the Object Petri Net family)を開発している．本研究プロジェクトでは，その一つとして，OPeN/WF(the Object Petri Net for Workflow)の開発を推進している．但し，本論文では，わかりやすさを優先し，OPeN 仕様のうち，一般

に EOS(Elementary Object System; 初等オブジェクトシステム)と呼ばれるレベルに相当する範囲のみ(OPN-0 レベル)を紹介するものとする．

さらに，本論文では，試作中の OPeN/WF 仕様のオブジェクト指向ペトリネットのためのワークフローエディタ，その実行を制御するワークフローエンジン，そして，複数のワークフロー参加者が携帯できるワークフローモニタに関して，その機能とアーキテクチャについて述べる．

ワークフローエディタは，ワークフローの分散開発を指向して，サーバ上に保管しているワークフロー記述をクライアント端末から編集する C/S 構成をとっており，一種の共同執筆機能として，複数開発者の間を調停するために版管理機能とチャット機能を備えている点に特徴を持つ．ペトリネットによるワークフロー表現では，一般に直観的な理解容易性に欠けると指摘されることもあるが，本研究プロジェクトでは，その欠点を補うにあたって，曖昧性や非厳密性を許容する表現をモデル記述言語に本質的に導入するという方法をとらず，エディタ等のツールで対応することを指向している．本稿執筆時には開発中であるが，エディタに一種のワークフローパターンを導入する機能，基本的なワークフローと例外フローを分離・結合する機能，粗粒度の記述から段階的に詳細化(refinement)していく機能を追加すべく更なる改良を続けている．

ワークフローエンジンは，サーバ上で実行され，サーバに保管されている複数のワークフロー記述の進行を管理する．ワークフローエンジンは，ワークフローの進行に伴い，コンピュータで実現されているサービスを呼び出す，あるいは，人間のワークフロー参加者へモニタ端末を通してタスク開始を指示しタスク完了の返答を待つ，といった機能を備えている．本稿執筆時には開発中であるが，事前にタスク遂行者へタスク開始時期の予定を送る，あるいは，タスク遂行に必要な情報を整理して提示する，といった事前通知機能を追加すべく更なる改良を続けている．

本論文では，続く第 2 章でオブジェクト指向ペトリネットの概要，第 3 章で本稿執筆時の全体的なアーキテクチャ，第 4 章で共同執筆/分散開発のためのワークフローエディタの機能，第 5 章でワークフローエンジンとモニタの機能について，それぞれ述べ，最後に第 6 章でまとめる．

2. オブジェクト指向ペトリネット

本研究では，ワークフローの表現としてオブジェクト指向ペトリネットを用いる．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る．オブジェクト指向ペトリネットにおいて，トークンには 2 通りあり，通常の P/T ネット(Place/Transition ネット)におけるトークンであるブラックトークンと，別のサブシステムを表現するオブジェクトネットへのリファレンスに相当するリファレンストークンである．そうしたサブネットには，ワークフローに関与するリソース（人的資源や，外部サービスを提供するサーバなど）を対応付けることができる．これによって，ワークフロー参加者やサービスを提供するサーバごとにワークフローを記述することが可能となる．

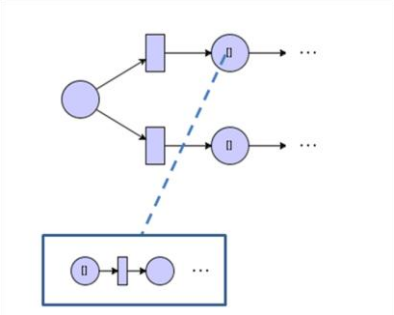

図 1 オブジェクト指向ペトリネット

本研究で用いるオブジェクトネットの概念は，EOS(初等オブジェクトシステム)[3]の概念に基づき以下の定義となる（詳細は，紙数の都合上，割愛する）．

定義 1　初等オブジェクトシステムにおけるオブジェクト指向ペトリネットの定義

オブジェクト指向ペトリネットは，$OS = (SN, ON_{m^0}, \rho, R_0)$ である．SN は，システムネットと呼ぶ．

ON_{m^0} は，P/T ネットの有限集合で，オブジェクトネットと呼ぶ．ρ は、オブジェクトネットのトランジションとシステムネットのトランジションの間の同期関係を表す．R_0 は，初期化マーキングである．

3. 全体的なアーキテクチャ

本論文では，ワークフローの参加者がワークフローを改善していく際に役立つ共同執筆ならびに分散開発の機能を備えたワークフローエディタと, ワークフローの遂行を管理するサーバであるワークフローエンジン, ならびに, エンジンと連携して，ワークフロー参加者との相互作用や，外部サービスとの相互作用を受け持つモニタ機能に関して紹介する．

その基本的なアーキテクチャは，C/S(クライアント/サーバ)方式であり，個々のリソースが担当するサブワークフローに対応するオブジェクト指向ペトリネットのクラス定義を蓄積補完し，さらにエンジンとしてその実行を管理するサーバ（WFMS: WorkFlow Management System）と，エディタ機能やモニタ機能をそなえたクライアントから構成される．

4. 共同執筆/分散開発のためのワークフローエディタの機能

ユーザは，Web ブラウザをつかってサーバにアクセスし，Web アプリケーションとして実現されたエディタによってブラウザ上で編集機能を利用することができる．エディタでは，基本的なグラフ編集機能（メニューからのノードのドラッグアンドペーストとアークの接続）によってペトリネットとしてワークフローを作成したり，そのペトリネットの一部にアノテーション（注釈；コメント）を付与することができる．エディタは，基本的な編集機能に加えて，さらに合意形成のための作業支援機構として以下の機能を提供している．

(1) チャット機能：分散開発している開発者同士でチャットによってリアルタイムに意思の疎通を図ることができるのとともに，そのやり取りの時系列的な記録を残すことができる

(2) ノード評価/注釈機能: 共同開発作業に参加するユーザは，各ノード(トランジションかプレース)および既に付与されているアノテーションに対し，更にアノテーションを加えることができる．エディタの左上のノード選択ボックスから「Comment」を選択してドラッグアンドドロップで追加することが可能である．

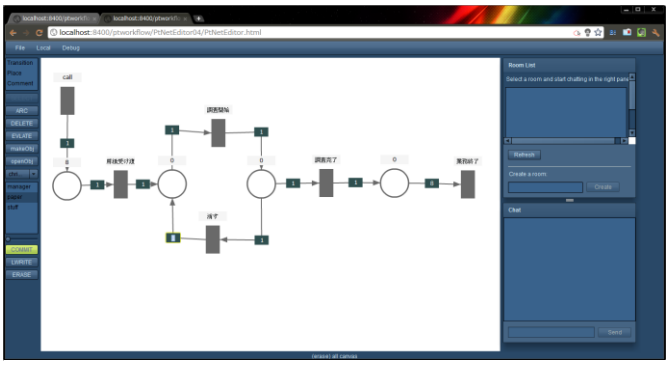

図 2 ワークフローエディタ（クライアント画面)

(3) 時系列バージョン管理 ：ユーザは他の開発者の操作過程が見えるよう，簡単な操作で旧バージョンを参照することができる．バージョン情報はファイルをアップロード（コミット）した際に自動的に設定しているタイムスタンプに基づいている．エディタ上では，左側のメニューの中段にあるスライダーを動かすことで旧バージョンを参照することが可能である．

現在は，このエディタに対し，更に，段階的詳細化機能，例外フロー分離機能，ワークフローパターン[4]の導入を支援する機能の構築を進めている．大規模で複雑なワークフローを作成するにあたっては，最初は粗い粒度で記述し，少しずつ詳細化していくことが望ましい．本研究プロジェクトでは，ワークフローの表現に曖昧さや直観的理解容易性を導入する代わりに，エディタの機能として段階的な詳細化の機能を提供することを選択した．また，ワークフローにおいて例外処理は極めて重要な要素であるが，ワークフローを複雑化し，わかりにくくする元凶となっている．例外フロー分離機は，例外処理と本来の処理と区別し，表示や編集の際に分離できる機能である．

5. ワークフローエンジンとモニタの機能

ワークフローエンジンはペトリネットの実行を管理するサーバであり，プロジェクト定義にしたがい，ワークフローエディタによって記述されたオブジェクト指向ペトリネットのクラス定義からインスタンスとしてのオブジェクトネット（とシステムネット）を生成して，同期関係の定義に基づいて各ネットを同期させながらトークンの遷移を進める．その進行状況は，サービスを提供する外部サーバに配置されたサービスモニタもしくは，各ワークフロー参加者が携行する Android 端末上のユーザモニタに通知され，必要に応じて，端末からの応答にもとづいて実

行を制御する．各タスク，ないしアクティビティの実行には時間が掛かるので，そのタイミングはモニタとの相互作用によって決定される．すなわち，タスクの開始はモニタに通知され，タスクの完了は，モニタからエンジンへと伝えられる．ユーザモニタは，ワークフローがサーバ上で実行されている処理状況を視覚化しワークフロー利用者に伝える役割も持つ．ワークフローエンジンは Java で，モニタは Flex(ActionAcript)で実装し，両者の連携には BlazeDS を利用している．

オブジェクト指向ペトリネットにおけるトランジションの発火条件は，システムネット並びにオブジェクトネットの集合において，同期関係にあるすべてのトランジションが発火可能であることであり，各トランジションはその全入力プレースに存在するトークンが，両者を接続するアークに記述されたガード条件（ブラックトークンしか持たない基本的な P/T ネットにおける多重度の概念をリファレンストークンに対応して拡張したもの）を満たす場合に限り発火可能となる．トランジションが発火すると対応するイベントが発生するか，もしくはタスクの遂行が開始される．イベントの発生には時間は要さないが，タスクの遂行には時間が掛かることが一般的である．イベントの発生もしくはタスクの遂行によって，リファレンストークンが参照しているオブジェクトの属性に対して指定された計算が行われる．そして，イベントの発生後，もしくはタスクの完了後には，出力アークに記載されたフィルタ記述（やはりブラックトークンしか持たない基本的な P/T ネットにおける多重度の概念をリファレンストークンに対応して拡張したもの）に基づいて接続するプレースへトークンが遷移する（トランジションに対応付けられた計算によっては，新たなオブジェクトが作り出され，それを参照するリファレンストークンが生成されることもある）．

したがって，サーバであるワークフローエンジンとクライアントであるモニタの連係動作の基本は，クラス定義に基づいてサーバがトークンを遷移させるたびに，次の発火可能トランジションのリストを，他のサーバ上にあるサービスモニタ，もしくは利用者が携行する Android 端末上のユーザモニタへ通知し，モニタからのタスク遂行通知を受け取って，次へトークンを遷移させることを繰り返すというものである．本研究プロジェクトは，基本的な連係動作を実現した段階であるが，次の計画として，タスクの遂行開始の前に，利用者へ前触れとなる通知（事前通知）を行う機能，タスク遂行に必要な情報を集めて，それを事前通知とともに利用者へ提供する機能の実現を進めている．これは，利用者にとって有益な情報を提供するにあたり，ワークフローが利用者のコンテキストを指し示すものとして活用できることを利用するものである．

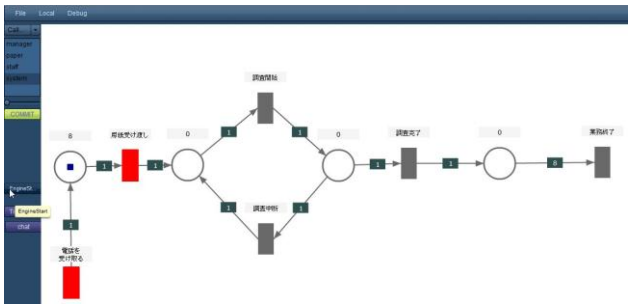

図 3 ユーザモニタ（クライアント）画面

6. おわりに

本論文では, オブジェクト指向ペトリネットに基づいてモジュール性を取り入れたワークフロー管理システムの，全体的なアーキテクチャ並びに，共同執筆/分散開発のためのワークフローエディタの機能と，ワークフローエンジンとクライアント Android 端末上のモニタ機能に関して述べた．エディタの段階的詳細化機能，エンジンとモニタによる事前通知機能等に関しては，現在開発中である．

参考文献

[1] W.M.P. van der Aalst and A.H.M. ter Hofstede: “YAWL: Yet Another Workflow Language,” QUT Technical Report, FIT-TR-2002-06, Queensland University of Technology, Brisbane, 2002.

[2] OMG: “Business Process Model And Notation (BPMN) , Version 2.0, ” formal/2011-01-03/, Release Date: January 2011, http://www.omg.org/spec/BPMN/2.0/PDF

[3] Rüdiger Valk: “Object Petri nets – Using the nets-within-nets paradigm, ” LNCS 3098, pp.819-848, Spriner, 2004.

[4] W.M.P. van der Aalst, A.H.M. ter Hofstede, B. Kiepuszewski, and A.P. Barros: “Workflow Patterns, ” QUT Technical report. FIT-TR-2002-02, Queensland University of Technology, Brisbane, 2002.