分散オブジェクトを用いてワークフローを実現するシステムモデル

5R－9

今野 和浩　久世 和資　北山 文彦
日本アイ・ビー・エム (株) 東京基礎研究所

1　はじめに

オフィスなどの環境におけるグループの協調作業を支援するため、グループウェア、あるいはComputer Supported Cooperative Work (CSCW)の考えに基づいて、各種のワークフロー管理システム製品が複数社から現在提供されている。ワークフロー管理システムとは、ビジネスプロセスをコンピュータ上の表現で定義し、その各プロセスアクティヴィティに対応するソフトウェアの実行順を制御することで、ビジネスプロセスの管理を行うシステムである。これらのシステムでは各アクティヴィティの実行順序や実行できる担当者の設定を容易に変更できるため、変化の速いビジネスプロセスの要求に柔軟に対応できる。

また、分散環境でのオブジェクト指向開発のための基盤として、CORBA規約に基づく分散オブジェクトの開発環境 (C++、Smalltalkなど) も各社から出されている。ワークフロー管理システムが対象とする環境は基本的に分散環境であるので、その上で使うアプリケーションをオブジェクト指向の手法で開発する場合に分散オブジェクトを使うのは自然な考えである。

しかし、既存のワークフロー管理システムは分散オブジェクトを用いて構築されたアプリケーションとの連携は基本的に考慮していないので、互いの想定するモデルに不整合が見られる。本論文では、ワークフロー管理システム (以下WFM) と分散オブジェクトを用いて構築したシステムの経験から問題点を指摘し、分散オブジェクトと結びつけた形のワークフロー実行モデルを提案する。

2　問題点

金融分野の支店業務システムのプロトタイプを、以下のような組み合せで構築した。

1. ビジネスオブジェクト (CORBA準拠の分散オブジェクト)
2. 各プロセスアクティヴィティに対応するGUIアプリケーション (上記分散オブジェクトを扱えるSmalltalk環境)
3. WFM (および、その必須前提として関係データベースシステム)

このプロトタイプを設計・実装していく間に判明した、WFMと分散オブジェクトの組み合せに起因する問題点を挙げると以下のようになる。

- 実行モデルの不整合: WFMがプロセスアクティヴィティという単位で大域的にフローを制御しているのに対し、分散オブジェクトはメッセージ送信の単位でLAN上で通信を行う。両者が混在する環境では計算の制御を何が握っているのか (握っているべきなのか) が不明確になりやすい。特に、ワークフローの環境には必ず人とのインタラクションが入るので、このことは設計上問題となる。
- WFMとアプリケーションの間の通信: WFMはアプリケーションとの間で、フロー制御のために終了 / 分岐条件などの判定のためのデータをやり取りする必要があるが、各アプリケーションが分散オブジェクトを用いて構築されているのに対し、WFMからは直接分散オブジェクトをアクセスできない。上記プロトタイプではこの通信のために、WFMが提供している共有変数空間 (共有ファイルシステムにより実現) を経由して間接的にデータの共有を行っている。
- 分散オブジェクトとWFMはそれぞれサーバを必要とし、さらにWFMは通常データベースシステムを利用するので多数のサーバが必要となり、管理上あるいはハードウェア資源の問題が起こる。

3　解決

既存のアプリケーションプログラムをWFMで結合して呼び出すのでなく、各プロセスアクティヴィティで呼び出されるアプリケーションを分散オブジェクトを用いて開発するのであれば、WFMが

Workflow System Model Using Distributed Objects.
KONNO Kazuhiro, KUSE Kazushi, KITAYAMA Fumihiko (IBM Research, Tokyo Research Laboratory)

各アプリケーションと全く独立のパッケージである必要はない。WFM の機能を、分散オブジェクトのネットワークとして以下のような枠組みで実装することを提案する。

1. 担当者に対応するクラス: ユーザが login するごとにそのマシンにインスタンスを生成する。ある瞬間にその担当者が実行可能なプロセスアクティヴィティのインスタンスを管理し(ワークリスト)、ユーザに一覧表示をして知らせる役目を持つ。
2. プロセスアクティヴィティに対応するクラス: フローの実行が開始されると最初のプロセスアクティヴィティに対応するインスタンスを生成する。アクティヴィティで起動するアプリケーションの情報などの他に次に起動されるアクティヴィティへの遷移条件 (分岐条件・終了条件) を持つ。各アクティヴィティのインスタンスは、そのアクティヴィティを実行できる担当者が login しているマシンへ次々と移動しながらアプリケーションの実行・遷移条件の評価を行い、フローを実現する。WFM サーバに集中していたフローの制御・担当者の割り当ての負荷がこれにより LAN 上に分散される。
3. 実行のトレース: 誰がいつプロセスアクティヴィティを実行したのか、あるフローがどこまで進んでいるのかなどの情報を管理することも WFM の重要な機能である。担当者クラス・アクティヴィティクラスと通信して実行状況の記録を行うオブジェクトがその役割を果たす。

このような枠組みにすることで、分散オブジェクトで組まれたアプリケーションと WFM とが直接通信できるようになって親和性が上がる他に、

- 担当者やアクティヴィティを抽象クラスとして用意することで、WFM の挙動を、継承やリフレクションにより柔軟かつ容易に変更・再利用できる。例えば、新しいアクティヴィティを実行できる複数の担当者が同時に login している際に、負荷の大小を見て柔軟な担当者割り当てを行うなど。
- 担当者クラス・アクティヴィティクラスのネットワークがオフィス組織や処理手続きの構造を表すので、このネットワークを業務の形態のフレームワークとして提供することで、各現場の事情に合わせて特化することが可能になる。

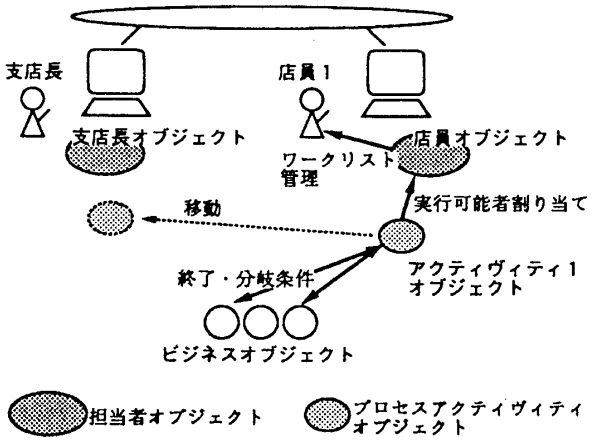

図 1: 提案する実行モデル

などの長所が得られる。また、WFM の機能の必要な部分は分散オブジェクトで実現されるので、既存の WFM のサーバおよびクライアントは不要となる。

このモデルをさらに押し進めると、ある担当者のオブジェクトとその人が実行可能なアクティヴィティのアプリケーション全てを実行単位として、その間をプロセスアクティヴィティオブジェクトが移動していくような形になる。ここに至って WFM は、その柔軟性を失わないまま、独立のパッケージではなく分散協調システムを構築するためのアプリケーションの構成要素となる。

4 まとめ

ワークフロー管理システムと分散オブジェクトによるアプリケーションの間のモデルの不整合を指摘し、ワークフローの機能を分散オブジェクトのフレームワークとして提供するモデルを提案した。これにより、両者の親和性が高まるほか、担当者、プロセスアクティヴィティなど、ワークフローシステム中の概念もオブジェクト指向による変更・再利用性の恩恵を受けることができるようになる。

参考文献

[1] Workflow Management Coalition. Glossary — A Workflow Management Coalition Specification, 1994.

[2] Ann Palermo and Scott McCready. Workflow Software: A Primer. In *Groupware '92*, pp. 155–159, 1992.