反復的な改善を含むソフトウェアプロセス制御システムの試作と考察*

3Q－3

斉藤 良和†　橋本 信也†　大木 幹雄†
日本工業大学§

1. はじめに

最近の開発工程モデルは，スパイラルモデルを代表例として，反復的な試作・評価・改善作業を含むことが一般的になっている．しかしながら，このような開発工程の形式的な表現モデルとして決定的なものは少ない．

現在まで実用に供されている図式表現の中で，開発プロセスの形式的な表現に用いることができるものとして，以下がある．

(1) データフロー図
(2) 状態遷移図
(3) ペトリネット

データフロー図は，作業の洗い出しと，その間で受け渡されるドキュメントを表現できる．状態遷移図は，作業の終了状態に応じて動的に次のどの動作を開始すべきか決定できるが，出力チャネルを複数記述する必要がある．ペトリネットは，作業の起動条件式を状態式の評価と起動式に分離する．並行作業をはじめとして柔軟に対応できるが，動作式は記述されない．

(1)～(3)は，それぞれ特徴があり，受け渡す情報は異なるが，オブジェクト同士の状態を監視し合い，あるオブジェクトの変化がそれに依存するオブジェクトの動作を決定するという点で類似している．この考え方と共通するものとして Observer モデルがある．そこで我々は，Observer モデルの考え方に沿って(1)～(3)の表現を統一化し，開発プロセスを一般して表現できるようにしたプロセス記述言語 UPFL（Unified Process Flow Language）を考案した．

2. UPFL の概要

2.1 UPFL の基本動作

UPFL は，作業中の状態を示す State-object(以後 S-object)と作業の関連，および作業開始・完了条件を示す Transition-object(以後 T-object)の二種類のノードからなる有向グラフ構造で，監視式によりプロセスの流れを制御するプロセス記述言語である．

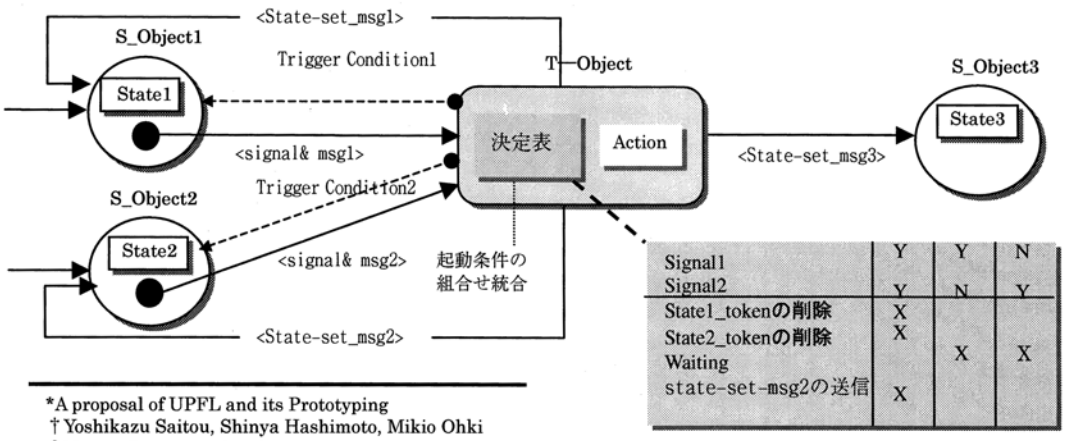
UPFL では，図 1 に示すように，関連がある T-object と S-object とが動的依存関係により，互いの変更を監視し合うということが基本的な動作となる．

T-object と S-object 間の実際の動作としては，まず，T-object 内にプログラムとして記述されている監視式が，各 S-object に送信される．疑似コードに変換された監視式は，入力 S-object を監視する．S-object は，その監視式に対する答えの値に変化が生じるごとに T-object に状態を表すシグナルを通知する．T-object は，決定表を参照することにより，この一つ，または複数のシグナルの内容から，起動すべきアクションを選び，作業終了や再作業開始の動作を実行する．例えば，作業が終了したら，トークンをセットするメッセージを次の S-object に送信し(S-set_msg3)，トークンを削除するメッセージを入力 S-object に送信する(S-set_msg1, S-set_msg2)．

2.2 UPFL の特徴

UPFL の特徴は以下の通りである．

(1) 作業の並行関係を明確にしながら，日程計算や



*A proposal of UPFL and its Prototyping
†Yoshikazu Saitou, Shinya Hashimoto, Mikio Ohki
§Nippon Institute of Technology

表 1. State-object の属性

<attribute>
State-object の名称
作業時間
作業完了監視条件
入力 Transition-object 名リスト
出力 Transition-object 名リスト
作業に要求される技能レベル
担当者の技能レベル
担当者名称
メタ作業監視条件
<method>

<attribute>
Transition-object の名称
<最早開始時間>
<最遅完了時間>
作業開始制約条件
入力 State-object 名リスト
出力 State-object 名リスト
再作業開始条件
再作業の特別アクション
メタ再作業アクション
<method>
決定表を用いたシグナルと動作定義

動的な変更によるプロセスの修正にも柔軟に対応できる．

(2) ペトリネットでは，トランジションに動作式は記述されず，前の作業すべてのトークンが到着するのを待つという動作のみで次の作業に移るので，複合する発火の条件動作は容易に表現できるが，OR 動作を表現するのは困難である 2)．しかし，UPFL では監視式により容易に表現できる．

2.3 試作システムの機能

前述の特徴を含む UPFL の試作システムの主な機能として以下がある．また，S-object の属性を表 1，T-object の属性を表 2 に示す．

(1) 日程計算

各作業の完了予定時間や，クリティカルパスを求める．手戻りが発生した場合にも，動的に再計算を行う．

(2) 監視式の評価

インタプリタ機能により，入力した監視式を構文解析して疑似コードを生成する．

(3) 評価結果による条件動作

決定表を参照することによって，S-object からのシグナルによる動作を決定する．

(4) 再作業時のプロセス修正

再作業が開始された場合，動的にプロセスの修正を行ったり，作業の追加を行ったりする．また，T-object に，並行的に開始する作業間の依存関係を持たせ，再作業が発生したときの作業の継続，あるいは中断した作業に連鎖して作業を中断する等の制御を行う．

3. 考察

UPFL では，再作業発生に対し動的に対応できる事は前述した．この再作業に関連する制御について以下のような動作を考慮する．

(1)作業の連鎖依存性についての制御

プロセスは，場合によっては T-object から複数の S-object に流れることもある．この際，S-object 間の動作として，一つの S-object の再作業が他のプロセスに影響しない独立型と再作業によって一方の作業も連鎖的に再作業にしなければいけない連鎖アボート型がある．連鎖アボート型には，例えば，データ構造を定義するプロセスとそのインターフェースを作成するプロセスのような関係が考えられる．この 2 種類の並列開始作業に対応することは，実際にプロセスを管理する上で有効に働くであろう．

(2)メタ制御による作業の再構成と動的な生成

UPFL では，再作業が起こった場合，動的にプロセスの修正，追加を行う可能性がある．この場合，プロセス変更になった原因をもとに再構成をする必要がある．このときプロセスの修正，追加手順，事例など，以前に起きているものは，知識データベースなどで資源化されているはずである．管理者は実際にそれらを参考にしながらプロセスを再構成するという手順をとる．このようなことを行うには，S-object を監視する作業プロセスとは異なるレベルのオブジェクトが必要になる．

4. まとめ

我々は，Observer モデルの考え方を拡張し，T-object と S-object に動的依存関係を持たせ，互いの変更を監視し合うという機構を持った UPFL を考案し，その動作をシミュレートするシステムを試作した．これによって再作業が生じる場合でも柔軟に対応できる機構の有効性が確認できた．

参考文献

1) 何克清，宮本衛市：ソフトウェアプロセスの基本制御構造，情報処理学会論文誌 Vol.33 No.11 pp.1414-1422 (1992)

2) 佐伯元司：ソフトウェアプロセスのモデル化へのネットの応用，情報処理学会論文誌 Vol34 No.6 pp718-730(1992)

3) 大木幹雄：動的依存関係を利用した分散オブジェクトの協調機構に関する提案，電子情報通信学会 信学技報 KBSE98-4 Vol.98 No.239 (1998)

4) 大木幹雄：分散協調動作モデリング言語Wed の基本概念と仕様記述，電子情報通信学会 信学技報 KBSE Vol.99 No.53 pp.21-28 (1999)