2M－5

進化・適応型プロセス支援環境の構築
－プロセスオブジェクト管理サブシステム－

村田 篤紀，東 基衛，吉田 武司
早稲田大学理工学部

1 概要

我々は，DREAM（Design, Refine, Evolve and Adapt paradigm）パラダイムに基づいた進化・適応型プロセス支援環境PEACE（Process Evolution and Adaptation CASE Environment）の構築を目指している．本論文では，PEACEで扱われるプロセスとプロダクトを管理するPOMSS（Process Object Management Sub-system）についての要求・概念設計について考察している．本研究の目的は，プロセスとプロダクトの進化を考慮に入れたそれらの管理システムの概念モデルを確立することであり，ソフトウェア構成管理（SCM）の観点に基づいてモデリングを行なった．

2 POMSSの役割

POMSSとはプロセスとプロダクトの論理的なもしくは物理的なrepositoryを表わすものであり，それを通してツールがデータ（オブジェクト）を共有したり，互いにコミュニケートしたりする．

ここで，ソフトウェア開発プロセスで出力されるオブジェクトは多数にのぼり，多様で複雑な構造を持ち，他のオブジェクトと複雑な関連をもっている．更にソフトウェア開発において変更は避けられないものであり，これらのオブジェクトは，任意の時間，異なったツールに共有され，複数のユーザにアクセスされる間，一貫性を持った状態で管理されなければならない．これはPOMSSのSCMならびにCASE repositoryとしての側面であり，一般的なエディタやCASEツールなどによってプロダクトが変更されたときの変更通知機能やバージョン管理，協調作業時の排他制御のような同期管理そしてトランザクション管理機構などを持つ．つまり，これらを扱うSCM機能をPOMSSのkernelとして実現すべきであることがわかる．

Evolutional and Adaptive Process Support Environment - Process Object Management Sub-System -
Atsunori Murata, Motoei Azuma, Takeshi Yoshida
School of Science and Engineering, Waseda University

一方，PEACE中でのプロセスの設計・進化ならびに適応は，DRESS（Design and Refinement Sub-system）・EASS（Evolution and Adaptation Sub-system）といった他のサブシステムもしくは人間の役割作業であり，POMSSはDRESSやEASSとインタラクトし，それらの要求に基づいて過去のオブジェクトを提供したり，設計・進化ならびに適応されたプロセス（オブジェクトとして扱っている）を格納したりする機能を持つ．

3 要求

以上から，POMSSには次の要求が課せられる．

(1) SCM機能を備えてなければならない．

(2) ソフトウェア開発は長期間にわたるため，プロダクト，プロセス，方法論そして組織など様々な変更（進化）の影響を受ける．また，組織固有の開発標準にも適用できなければならない．よってこれらに対処するために，カスタマイズや拡張が可能でなければならない．

(3) 過去のプロセスの利用やカスタマイズに関して，それらの評価と査定が重要になる．プロセス管理の最終目的は，生産性とプロダクト品質の向上であるため，プロダクトの品質とその適用プロセスを組み合わせて保持できなければならない．つまり，プロジェクトのプロセスモデルのプロセス実行中や後での評価のためにコンテキストに依存した形でそれらを保存しておかなければならない．

4 概念モデル

POMSSは，プロダクトを現わすProductObject，プロセスを現わすProcessObjectそしてプロジェクト全体を現わすProjectObjectから構成され，SCM機構の実現の為に，基本的にすべてのオブジェクトがバー

ジョンを保持している．

ProductObjectは個々のプロダクトに対する属性とアクセス・更新メソッドを保持している（図１参照）．「部分オブジェクト」には，当該オブジェクトが集合物であった場合の下位の構成オブジェクトが入り，構成オブジェクト間での一貫性を保つために使用される．集合物でない場合はNILが入ることになる．「依存物」には，ヘッダファイル等の内容物自体が依存しているものやプロセスを通して依存しているオブジェクトの集合が入れられる．これを用いて変更通知機構が実現される．「状態」はbaselineを考慮しており，テストやソフトウェア品質保証が終わってリリース後になれば他からの使用を認めることに使われる．また，プロダクトは何等かのプロセスによって生成されるため，導出プロセスという属性中にProcessObjectのインスタンスが保持される．

ProductObject

名前
内容物
所有者
部分オブジェクト
アクセス権
ログ
導出プロセス
依存物（内容物内での依存物や導出プロセスの入力物）
プロダクト自身に関する知識
状態（開発中，テスト中，リリース後等）
バージョンログ（バージョンのインスタンスの集合）

図１：ProductObjectクラスのインスタンスの構造

ProcessObjectはプロセス自身に関する属性・メソッドを保持している（図２参照）．「実施者」は誰によってなされたかを示すものであり，「タスク」とは，なされるべきことの記述である．また，どのように行なったかについては「メソドロジ」に，なぜそのように行なったかという記述は「プロセスに関する知識」に保持される．「メソドロジ」には，ツールの使用も含まれる．つまり，POMSS中ではツールの使用はプロセスの一部として考慮され，ツールの変更や追加はプロセスの変更としてモデル化される．

ProcessObject

名前
入力物のset
出力物のset
実施者
プロセス自身に関する知識
メソドロジ（タスク実行のための知識）
タスク集合
コマンドスクリプト（or ツールの起動コマンド）
バージョンログ（バージョンのインスタンスの集合）

図２：ProcessObjectクラスのインスタンスの構造

ProjectObjectクラスは，プロジェクト中に現れた複雑なネットワークを構成しているProductObjectと関連するProcessObjectすべてを含むオブジェクトのクラスである．ここでは，組織に関する記述やプロセスの流れが表現される．これらは，ProductObjectならびにProcessObjectの集合でモデル化され，プロジェクト全体のバージョン並びにメトリクスを保持している．例えば，要員の追加やツールの追加・置き換えなどは，プロジェクトの一つのバージョンとして保持される（結果的にはそれがプロセスにも反映されることになる）．

以上のようにモデリングされたオブジェクトは，過去の事例として他のサブシステムに提供される．また，サブシステムからの要求によってプロセス並びにプロダクトのオブジェクトの進化が起こったときには，それぞれのクラスをサブクラス化することによって各オブジェクトを詳細化しながら対応し，新規オブジェクトを格納する事になる．また，それらは個々の属性の変更をもとにバージョニングされているため，プロダクト並びにプロセスの進化をバージョンを追う事でみる事が出来る機能を提供する．

5 今後の課題

まず，この概念モデルはまだリファインの途中であるため，更なるモデリングを行う．また，動的な側面についての記述が明確になってないためこれらの記述を行ってから，詳細設計へと移る予定である．

一方，POMSS自体も進化・適応型にするためには何等かのメタなモデルが必要であり，メタモデルのモデリングも随時おこなっていくつもりである．