社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2005－SE－147 (8)
2005／3／17

PReP モデルに基づくソフトウェアプロセス運用支援ツールの試作

本村拓也† 上野秀剛† 大岡徹也† 木村隆洋†
秋永和宏† 大杉直樹† 田中康‡ 飯田元†

†奈良先端科学技術大学院大学 情報科学研究科

E-mail : {takuy-mo, hideta-u, tetsuy-o, taka-kim, tomohi-a, naoki-o}@is.naist.jp
iida@itc.naist.jp

‡ソニー株式会社

E-mail :Yasushi.Tanaka@jp.sony.com

本稿では，PReP (Product Relationship Process) モデルに基づく，ソフトウェアプロセス運用支援ツールの試作について紹介する．PReP モデルは，ソフトウェア開発プロジェクトのライフサイクル中に存在する成果物間の関連に着目したソフトウェア開発プロセスモデルである．PReP モデルにおいては，ダイヤグラムによる成果物間の関連記述をもとに，開発プロジェクトで利用可能なスケジュール情報を導出することが可能である．我々は，作図ツール (Microsoft Visio) を用いて成果物間の関連を記述し，そこから，プロジェクト管理ツール (Microsoft Project) で読み込み可能な形式のスケジュールデータを生成することの出来る一連のツールを開発した．本ツールを用いることにより，PReP モデルに基づくソフトウェアプロセスの記述と実プロジェクトへの運用が容易になり，開発現場における PReP モデルの導入を促進できると期待される．

A Toolset for Generation of Project Schedule from Process Description based on PReP Model

Takuya Motomura †Hidetake Uwano †Tetsuya Ohoka † Takahiro Kimura †
Tomohiro Akinaga † Naoki Ohsugi † Yasushi Tanaka †‡ Hajimu Iida †

†Graduate School of Information Science, Nara Institute of Science and Technology

E-mail : {takuy-mo, hideta-u, tetsuy-o, taka-kim, tomohi-a, naoki-o}@is.naist.jp
iida@itc.naist.jp

‡Sony Corporation

E-mail :Yasushi.Tanaka@jp.sony.com

Abstract This paper describes prototype development of the employment supportive tools of the software process based on a *Product Relationship Process (PReP) model*. PReP model is a modeling method of a software development process that mainly focuses to the relation of artifacts in the lifecycle of a software development project. Relation diagram in PReP model can be drawn with an existing drawing tool (Microsoft Visio), and developed tools generate datafile for an existing project management tool (Microsoft Project). Ease of use of these tools will be strong help for description and employment of PReP model in a real project, and introduction the PReP model to development organizations will be accelerated.

1. はじめに

ソフトウェアに対する要求は，近年，急速な高まりを見せている．様々な分野領域それぞれにおいて，問題解決のためにより高度かつ複雑な処理を，より効率よく行うことができるソフトウェアが望まれている．こうしたなかで，ソフトウェア開発を行う企業にとっては，製品となるソフトウェアの品質が市場競争における重要な要素となっており[6]，ソフトウェア開発プロジェクト自体に関する開発プロセスの改善が重要な課題となっている[8]．プロセス改善および，改善による生産性や信頼性の向上を意図した試みの一つとして，ソフトウェアプロセスのモデル化がある．

ソフトウェアプロセス改善のための参照モデルとして各所で幅広く利用されている CMM (Capability Maturity Model) の定義によると，ソフトウェアプロセスとは「ソフトウェア並びに関連する生産物を開発，保守するために使われる一連の活動、手法、プラクティス，変換」と定義されている[2]．ソフトウェア開発プロセスのモデル化とは，特定の観点と抽象化レベルなど一定

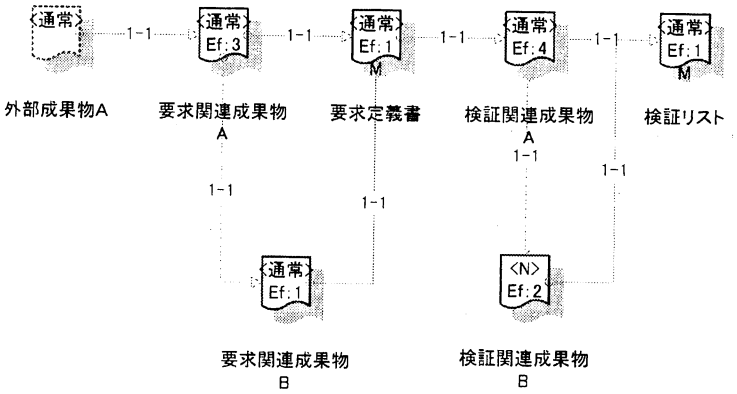

図1 構造モデルの記述例

の方策に従い，開発プロセス全体または一部を抽象化して記述することであり，開発時における進捗など，作業状況を把握，確認するための様々な指標を適切に得るために重要な活動である．また，開発プロジェクトの計画，終了後のプロセス改善活動や再利用を効果的に行うためなどにも非常に重要である[3]．

しかし，開発プロセスの適切なモデル化とその運用には一般に，利用するモデル化方法をよく理解した上で，煩雑な作業を，モデルの定義に沿って的確に実践する必要がある．モデル化作業が，利用するモデルの定義に沿って的確に行われなければ，モデルの特性から得られるはずの利点が失われることになり，開発のスケジュールなどに悪影響が及ぶことは明らかである．そのために，実際の開発現場においてプロセスモデルを有効に活用するためには，ツールによる支援が欠かせない．

PReP モデル[1]は，ソフトウェア開発におけるプロジェクト管理とプロセス改善を目的として，奈良先端科学技術大学院大学とソニー株式会社により提案された，成果物間の関連に基づいたプロセスモデルである．我々は，開発現場における PReP モデルの導入促進を支援するために，PReP モデルによるプロセスの記述や，それによる現場でのプロジェクト管理への活用を支援するための支援ツールの試作を行った．本稿では，PReP モデルによるプロジェクトプロセスの記述を容易に行うために行った，Microsoft(以下 MS と略記する)社の作図ツール Visio への拡張や，プロセスのダイヤグラムから，MS 社の MS Project で読み込み可能な形式のスケジュールデータを生成するコンバータの試作について述べる．

以下，本稿の構成について述べる．2. で PReP モデルについて概説する．3. で作成した支援ツールについて述べ．4. でその使用例を示し，5. でまとめと今後の課題を述べる．

2. PReP モデル

本節では PReP モデル全体の構成と記述形式について概説する．PReP モデルに関するより詳細な情報は[1]を参照されたい．

2.1 記述の観点と特徴

PReP モデルは，工程モデル，構造モデル，作業モデルの3つのサブモデルによって構成された，成果物指向のプロセスモデルである．PReP モデルでは，開発プロセス全体で行われる実際の活動の順序や十分条件などの関係を，成果物間の関連で記述する．PReP モデルにおける成果物とは，「各作業者に割り当てられた作業の成果としての実体であり，かつ振る舞いを持つもの」と定義され，例えば，要件や仕様，実行コードなどが当てはまる．また，成果物間の関連とは，開発プロセスへの最初の入力となる成果物から最終成果物が作られるまでの，個々の成果物間の入出力関係を指す[1]．個々の成果物を作成するための手続きについては，組織が現在使っている，既存の標準化された手続きを用いることができる．

過去に提案されたプロセスのモデル化方法の中で，PReP モデルと関連する代表的な研究例としては，Humphrey らの提案した EPMs (Entity Process Models) がある[3]．EPMs は，開発ライフサイクル中に現れる実体である，エンティティに着目したモデル化方法である．EPMs はエンティティの状態遷移に着目しており，開発プロセス全体の詳細な記述を試みると，記述の複雑さが増大してしまうという問題があった．成果物はエンティティの一種であり，その点では PReP モデルは EPMs と同じ点に着目をしているが，成果物の状態遷移は（必要であれば）作業モデル内に記述することとし，構造モデル自体には成果物の関連のみに焦点をあてているため，EPMs にくらべて記述が容易で理解しやすいものとなっている．

開発プロセスを成果物間の関連を用いて記述する方式は，従来型の，タスク間の順序関係による記述と比べて，開発プロジェクト自体の大幅な変更が無い限り変化しにくいという性質を持ち，再利用性や管理容易性などに優れている．また一方で，成果物の作成に関しては，従来通りの，局所的なプロセスの記述と定義に有用性が認められている，成果物の作成に必要

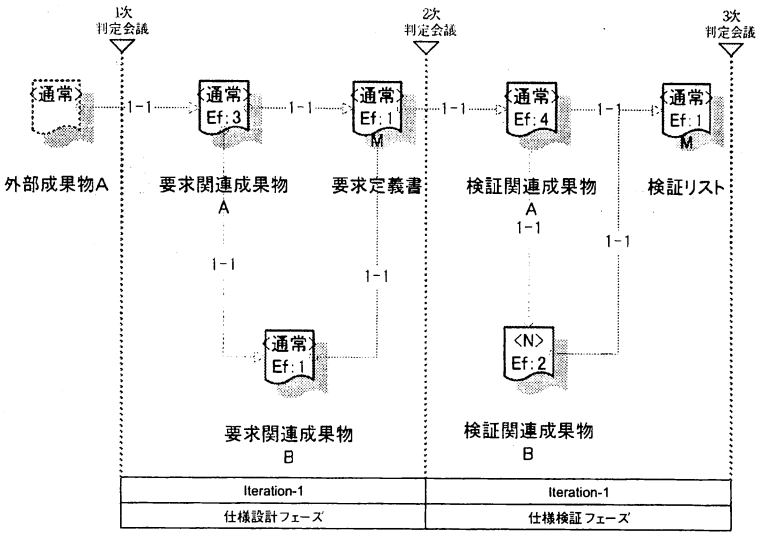

図2:工程モデルの記述例

な作業である，タスクからの観点による記述を用いることができる．

2.2 作業モデル

作業モデルは，成果物の作成に必要な具体的な作業手順情報を記述する部分であるが，現バージョンのPReP では特に記述方式は定めておらず，ETVX[4]やIDEF0[7]などの一般的なフロー図や，Daibutsu-Den形式[5]の XML 記述など，既存のタスク指向プロセスモデルの利用を想定している．

2.3 構造モデル

構造モデルはプロセスに含まれる成果物とそれらの間の関連を図で記述したもので，組織の標準プロセスを定義し，その適用を行うための記述方式である．また，開発完了後には構造モデルの記述を開発プロセスの評価と改善のために利用することもできる．各成果物に対する作業モデルの指定は，構造モデルを記述する際に行う．

成果物の種類

成果物には「通常成果物」と「並列成果物」の2種類が定義される（図1）．通常成果物とは，1つの情報単位から構成される成果物である．並列成果物とは，同じ種類の独立した複数の情報単位から構成される成果物である．ここで情報単位とは，成果物を構成する情報の単位又は型のことであり，例えば，モジュール単位，機能単位などが当てはまる．

成果物には様々な属性が定義されている．属性には名称やデータ形式，マイルストーン成果物（2.4 で詳細に説明）を指定するためのフラグなどがある．

関連の種類

成果物間には「入力関連」，「同期関連」，「入力と同期の合成関連」の3種類の関連のいずれかを指定可能である（図1）．入力関連は，成果物間に引かれた単方向実線の矢印で示され，矢印の始点側の成果物が終点側の成果物作成開始に必要な入力となる関連を表す．同期関連は，双方向点線の矢印で示され，矢印の両側の成果物が情報を交換し合い，同期をとって並行して作成される関連を表す．

入力と同期の合成関連は単方向点線の矢印で示され，入力関連の場合と同様に始点側にある成果物の情報を入力として終点側の成果物が作成され，かつ成果物間で同期をとって並行して作成される関係を表す．つまり，始点と終点の成果物は，並行して作成してよいが，始点の成果物が完成しないと，終点の成果物の作成も完了できない．

表記法

構造モデルの記述例を図1に示す．図中に示すように，成果物はドキュメントを表す図形オブジェクト(アイコン)で表現されており，その下に表示されている文字列は，成果物の名称である．また，アイコン内には，通常成果物と並列成果物の区別（<通常>あるいは<N>として表記）や，その成果物の作成に必要な工数（Ef:の後ろに表記）が示される．また，マイルストーン成果物のアイコンには記号 M が付与される．

関連の矢印上の"1-1"などの表記は，始点側と終点側の成果物間の関連に設定された多重度を表している．記号「n」を用いて一般的な多重性を指定できるが，実プロジェクトを記述する際にはプロジェクトの目的や制約条件に応じて，"正整数-正整数"といった具体的な形で表記をすることも許される．

図1に示されたプロセス記述の内容について具体的に説明する．まず，要求関連成果物AとBの間が入力と同期の合成関連，検証関連成果物AとBの間が同期

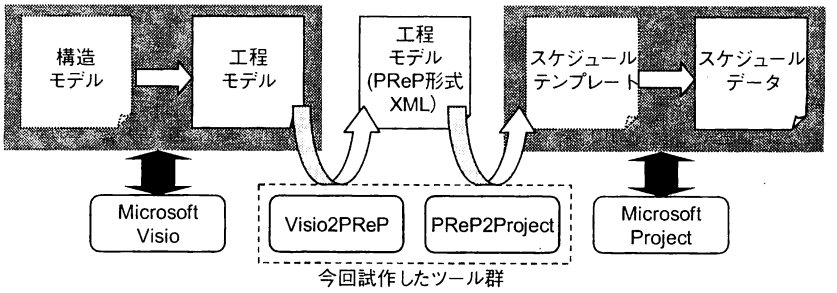

図3 PRePモデル利用時の作業フローとツール群の位置づけ

関連として定義されており，その他の関連は全て単純な入力関連である．

プロセスの最初の入力として外部成果物Aが利用され，要求関連成果物 A が作成される．要求関連作生物BはAと並行した開発が許される．Bの方が工数は少ないが，入力と同期の合成関連の定義より，Aの作成完了まではBの作成を完了できない．要求関連成果物 A, B の作成完了後に，これらを参照して要求定義書が作成される．

次に，要求定義書を入力として検証関連成果物A, Bが作成される．Bは並列成果物であり<N>個の情報単位で構成されることがアイコンに示されている．A と B の間には同期関連が存在し，並行して開発可能である．

検証関連成果物AとBの作成完了後それを入力として，検証リストが作成され，プロセス全体が完了する．

なお，外部成果物とは，他プロジェクトで作成が行われている，あるいは既に作成がされた，プロジェクトにおける管理対象外の外部から入力される成果物である．

2.4 工程モデル

工程モデルとは，構造モデルに対して開発のライフサイクルを定義し，実際のスケジュールへ展開を行った記述方式である．図2に示した工程モデルの記述例を用いて順に説明を行う．

まず，構造モデル記述に対して，マイルストーン成果物の指定を行う．工程（フェーズ）の終わりをマイルストーンと呼び，工程内の作業の検証をおこなう各種会議体などが設けられる．マイルストーンにおいて検証対象として指定された成果物を特にマイルストーン成果物と呼ぶ．構造モデルの記述に対してマイルストーン成果物と，それがどの工程に属するのかを指定することで工程の区切りが特定され，ライフサイクルが複数のフェーズの系列として構成可能となる．

図2においては「仕様設計フェーズ」と「仕様検証フェーズ」の2工程が存在し，要求定義書と検証リストがマイルストーン成果物と特定されている（記号「M」が付与されている）．

さらに，各フェーズに対しては反復（イテレーション）と呼ばれる工程の繰り返しを定義することができ，同じ構造モデルから，プロジェクトの特性に合致した反復回数の異なる工程モデルを得ることが可能である．なお，図2の例では簡単のために，いずれのフェーズにおいて反復回数は1回としている．

3. PReP支援ツール

PReP モデルの記述方式は，2 節で紹介した理解性の高いダイヤグラムを用いているが，独自に定義されたものであるため，実際の開発現場で PReP モデルに基づいたプロセス記述を行い，かつ，それに従ってプロジェクトを実行するには，次の二つの機能を提供することが特に重要である．

- PReP 形式のダイヤグラムを容易に記述するための機能
- PReP 形式のプロセス記述を元に，実際のプロジェクト運用（スケジュール管理）を行う機能

われわれは，これらの機能を実現するツールとして，多くの組織で利用されている既存パッケージ（作図ツール MS Visio とプロジェクト管理ツール MS Project）をそれぞれ活用するアプローチを採用し，これらのパッケージでPRePモデルの記述を活用できるようにするための2つの支援ツールの試作を行った．本節ではその概要について述べ，より詳細な実装内容と利用方法については4節において述べる．

3.1 ツール群全体の構成

図3に，支援ツール試作時に想定したPRePモデルを利用する際の作業手順を，実際にツールで支援を行う箇所とあわせて示す．まず作図用ソフトウェアMS Visio 2003 を利用し，構造モデル，さらにはそれを元にした工程モデルを記述する．Visio による記述ではPRePモデル記述用に作成した「ステンシル」を用いる．ステンシルとはMS Visio上で，目的に合わせて独自に定義することのできる，図面を作成する際の要素となる図形オブジェクトの雛形である．

次に，構造モデルを記述したMS Visio 形式のファイルに対して，第1の変換ツールであるVisio2PReP を使用し，あらかじめ PReP モデルの定義に従って作成されたDTD（XML 文書の構造定義）に基づいたXMLフォーマットへ変換を行う．この時点で得られるファイ

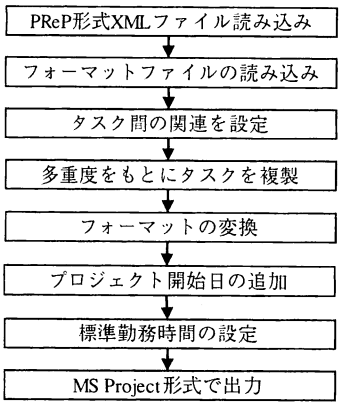

図4 PReP2Projectの内部処理手順

ルをここではPReP形式のXMLファイルと呼ぶこととする，次に，第2の変換ツールであるPReP2Projectを使用して，PReP形式のXMLファイルをスケジュール管理ソフトウェアMS Project 2003形式に変換する．以下，各ツールについて説明する．

3.2 Visio2PReP

Visio2PRePは，MS Visio上で記述した構造モデルのファイルを入力とし，PReP形式のXMLファイルを生成する．

処理概要

MS Visioは構造モデル図の記述に関して利便性が高いが,その出力ファイルには構造モデルを記述する際に用いたステンシルの座標、サイズなどの情報のみが記述されており，PReP モデルで定義されている構造モデルに関する意味的な情報は直接には含まれていない．そのため,構造モデルの再利用，工程モデルへの展開を行おうとする場合において不都合が生じる．

本ツールは，MS Visio 上で記述した構造モデルのファイル中に含まれる情報から，成果物の名称と種類，成果物間の依存関係と作成順序，スケジュール情報といった意味情報を抽出あるいは算出し，それらの情報をもとにPReP形式のXMLファイルを生成する．

実装上の特徴

本ツールは，オブジェクト指向スクリプト言語Rubyで記述されており，XML ファイル生成のためには，Rubyに標準で備えられたREXML[9]というXPathに対応した高機能 XML パーサを用いている．また，Ruby インタプリタをインストールしていない環境でも動作するように，exerb[10]を使用してWindows実行形式ファイルに変換をしている．利用する際の負荷も少なく，MS Visioで作成したファイルを本ツールのアイコンにドラッグアンドドロップすることで，PReP 形式の XML ファイルの生成を完了できる．

3.3 PReP2Project

PReP2Projectは，3．2で説明したVisio2PRePが出力するPReP 形式の XML ファイルを入力として，MS Project 形式のファイルを生成する．

処理概要

本ツールで行う処理手順を図4に示す．本ツールは入力となるPReP 形式の XML ファイルの記述内容に基づきMS Project で作業管理を行うための情報を以下の手順で生成する，

1) まずPReP モデルで定義された各成果物作成のための作業をMS Projectのタスク（図6の画面表示における，"タスク名"の部分に相当）として生成する．
2) 次に，成果物間に定義された関連の情報を元に，タスクの前後関係を設定する．また，それぞれの関連に指定されている多重度に従って，タスクを複製して，フォーマットの変換を行う．
3) 最後にPReP モデルでは定義されていない，プロジェクトの開始日や標準勤務時間などのスケジュールに必要な情報にデフォルトの値を追加し，MS Projectが読み込むことの出来る形式のXMLファイル（以降ではProject形式のXMLファイルと呼ぶ）を出力する．

実装上の特徴

本ツールはJavaで書かれたGUIを備えたツールであり，Java の動作する様々な環境において動作可能である．また，本ツールではPReP形式のXMLからスケジュール管理ツールのフォーマットへ変換するための変換ルールを，XSLT 形式で記述している．したがって，スケジュール管理ツールの仕様変更などに対して柔軟に対応することができる．本稿では前述の通りスケジュール管理ツールとして，MS Project 2003を使用し，その動作を確認している．

なお，現バージョンでは，フェーズの区切り指定を認識させるために，フェーズ終端にダミーの成果物を配置し，各マイルストーンからの入力関連を指定する必要がある．

4. 支援ツールの使用例

本節では，先に例として用いた図1をもとに，3節で述べた構造モデルの記述からスケジュールの作成までの一連の動作を通じて，支援ツール利用の詳細について説明する．

4.1 構造モデルと工程モデルの記述

まず，MS Visioを用い，構造モデルと工程モデルの記述を行う．このときPRePモデルの定義に基づいてあらかじめ作成したステンシルを用いる．また，モデル中において成果物に定義されている様々な属性値を設定することも可能である．図5にVisio上で成果物の属性を設定する際のダイアログの例を示す．成果物の属性値のうち現バージョンのPReP 支援ツール群が実際

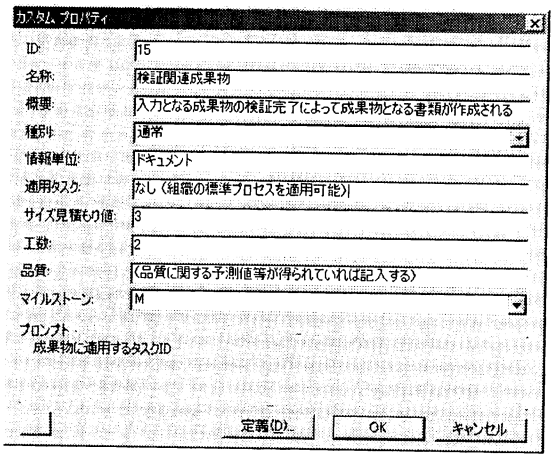

図5 :Visioでの成果物属性設定ダイアログ

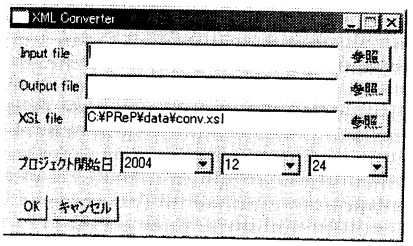

図 6 :PReP2Projectのパラメータ指定ダイアログ

に利用するものは，ID，名称，概要，種別，工数，マイルストーンであり，その他については今後対応予定である．

4.2 PReP形式XMLファイルとスケジュールの生成

次に，構造モデルの記述を保存した Visio のファイルを，Visio2PReP に入力し，PReP 形式の XML ファイルを出力として得る．さらに，PReP2Project を起動し，PReP 形式の XML ファイルを入力として，プロジェクト開始日を指定し，MS Project 形式の XML ファイルを生成する．図 6 に PReP2Project の GUI を示す．3．3 で述べたように，異なるスケジュール管理ツール毎に適した変換ルールを適用できるように，XSL ファイルを用いており，デフォルトでは MS Project 2003 用の変換ルールを記述した XSL ファイルが指定される．

最後に，生成された Project 形式の XML ファイルを MS Project 上で開き，稼働日や稼働時間など，標準勤務時間に関する設定を行い，スケジュール作成を完了する．こうして作成されたスケジュール情報を用いて，MS Project の機能により，PReP モデルに基づいた開発プロジェクトの運用を行うことができる．

図 7 に，図 1から得られた変換後のファイルを MS Project で開いたものを示す．この例では，プロジェクト開始日を2月 20日，「検証関連成果物 B」の並列成果物の情報単位数を N=3 として展開している．MS Project のスケジュール表示方式はガントチャートをもとにしている．図 1の構造モデルの記述に従い，成果物の名称，作成に必要な工数，関連などの情報が正確にスケジュール中に反映されていることがわかる．

なお，外部成果物に関しては既に外部のプロジェクトで開発が完了しているとみなされるため，製作日数は 0日となっている． また，工程モデルにおけるフェーズと反復については現バージョンでは完全には対応しておらず，図 7中にあるように擬似的にフェーズの区切りを表現する成果物（工数＝ゼロ）をダミーとして挿入することで実現している．反復については MS Project を用いて手動で反復回数分の複製を行う必要がある．

5. 考察

ある組織に PReP モデルのように新たなプロセスモデルの導入を推進する場合，モデルの適切さだけではなく，現場スタッフによるプロセス記述やそれに基づいたプロジェクト運用を支援する機構が不可欠である．また，支援機構を提供するにあたっては，導入にあたっての様々なコストの観点から，従来の環境との親和性が高いことが強く要求される．

ソフトウェアの開発プロセスを支援する目的で行われている研究で，PSEE (Process-Centered Software Engineering Environments)に関するものがある．PSEE とは，プロセスの支援を中心として，開発時において生じる様々な作業に対する制御や支援を統合した環境である．過去に様々な支援方法を持つ PSEE が提案されており，たとえば，プロセスのシミュレーションを提供するもの，自動化を行うもの，変更を支援するものなど多数のものがある[11]．しかし，実際の開発現場に導入され定着した例は見られない．その理由は導入への文化的敷居が高いことや，導入に成功しても開発技術や組織の変化に追従できないために使われなくなってしまったことなどが考えられる．

開発における新しい方法論の受け入れにはツールの有用性や使いやすさが重要な要因となる[12]．また，開発者はその利点がはっきりとわかるまで，新しい開発環境に移ることに抵抗しがちである[11]．PReP ツール群は，環境自体を提供するものではなく，開発プロセスの支援に利用することの出来る既存の比較的広いユーザ層を持つ高機能なアプリケーションソフトウェアを連携させた支援を可能とする．またそのため，導入コストの面でも負担が少なくすむと考えられ，今後の PReP モデルの運用と普及にあたり，強力な支援手段であると考えられる．

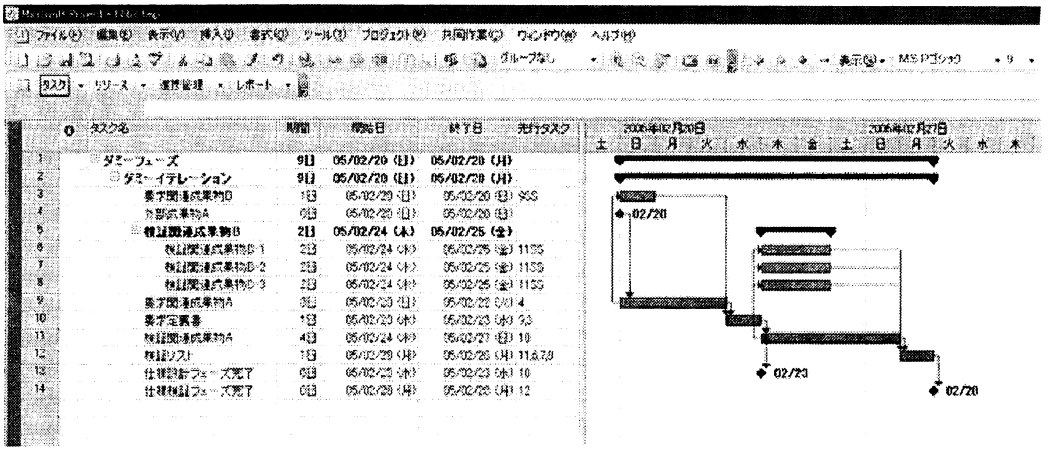

図 7 PReP2Projectによる変換後のファイル

6. まとめ

本稿では，ソフトウェア開発におけるプロジェクト管理とプロセス改善を目的としたソフトウェア開発プロセスのモデル化方法である，PReP モデルに基づく，ソフトウェアプロセス運用支援ツールの試作について紹介した．現在，実際に本ツールを活用したPReP モデルによるソフトウェア開発プロセスの運用が実企業の現場にて進められつつある．

今後，工程モデルにおけるフェーズとイテレーションの記述への完全な対応を予定している．また，より本格な運用支援のためには，本ツール群をバックグラウンドで実行することで自動的にプロセス記述を変換したり，Visio や Project のプラグインとして実装することでプロセス記述を直接やりとり可能であるような形態に変更することで一連の作業手順を隠蔽し，運用時のオーバーヘッドを軽減することが考えられる．さらに，PReP 形式の XML ファイルをスケジュール管理以外の目的（たとえば構成管理や品質管理，コスト予測など）にも活用し，より幅広い支援を提供することなどを予定している．

参考文献

[1] 田中 康，飯田 元，松本 健一，"成果物間の関連に着目した開発プロセスのモデルPReP"，情報処理学会論文誌，Vol. 46, No. 5, 2005 (to appear).

[2] M. Paulk, B. Curtis, M. Chrissis, C. Weber, "Capability Maturity Model for Software (Version 1.1)", CMU/SEI-93-TR-024, 1993.

[3] Watts S. Humphrey, Marc I. Kellner, "Software Process Modeling Principles of Entity Process Models", CMU/SEI-89-TR-002, 1989.

[4] Radice, R. , "A Programming process architecture, IBM Systems Journal", 24, no. 2. 1985.

[5] Hajimu Iida and Yasushi Tanaka, "A Compositional Process Pattern Framework for Component-based Process Modeling Assistance", Proceedings of the 1st Workshop on Software Development Patterns (SDPP'02), Technical Report TUM-I0213, Munich University of Technology, pp.57-64, 12, 2003.

[6] S. H. Kan, V.R. Basili and L. N Shaprio, "Software quality: an overview from the perspective of total quality management", IBM System Journal, Vol.33, No.1, pp.4-19, 1994.

[7] David A. Marca and Clement L. McGowan, "SADT : Structured Analysis and Design Techniques", McGraw-Hill, New York, NY, 1988

[8] 赤坂幸彦, "プロセス改善 成功と危機", SPI Symposium 2001 Fall :日本版ソフトウェアプロセス改善を考える, 2001.

[9] http://www.germane-software.com/software/rexml/

[10] http://exerb.sourceforge.jp/

[11] Alfonso Fuggetta, Alexander Wolf 編，岸田孝一 監修，坂本啓司，中小路久美子 監訳, "ソフトウェアプロセスのトレンド", 海文堂, pp.47-51, 1997.

[12] K. Riemenschneiber, C. Hardgrave and D. Davis, "Explaining Software Developer Acceptance of Methodologies: A comparison of Five Theoretical Models", IEEE Trans. Softw. Eng., Vol.28, No.12, pp.1135-1145, Dec 2002.