

プロセス資産の再利用を支援するプロセスモデルの提案と それに基づくプロセス中心型開発環境の試作

後藤 徹平[†] 飯田 元^{††} 松本 健一[†] 田中 康^{†‡}

[†] 奈良先端科学技術大学院大学情報科学研究科 〒630-0192 けいはんな学研都市

^{††} 奈良先端科学技術大学院大学情報科学センター 〒630-0192 けいはんな学研都市

[‡] ソニー(株) ネットワーク&ソフトウェアテクノロジーセンター 〒108-6201 東京都港区港南 2-15-3

E-mail: [†] {tepei-g, matumoto}@is.aist-nara.ac.jp, ^{††} iida@itc.aist-nara.ac.jp, [‡] Yasushi.Tanaka@jp.sony.com

あらまし 本報告では、コンポーネント化されたプロセス資産を用いることでソフトウェアプロセスの再利用を支援するプロセスモデルの提案と、そのプロセスモデルに基づいてプロジェクトの計画を策定し、進捗を管理するプロセス中心型環境の設計について述べる。提案するモデルおよびシステムは、CMM/CMMI レベル3に基づいたプロセス改善活動において重要な課題とされている、組織のプロセス資産の記述や再利用を重点的に支援することが可能である。

キーワード CMM, CMMI, ソフトウェアプロセス, プロセスモデリング, プロセス中心型開発環境

Software Process Model and Process Centered Environment for Process Assets Reuse

Teppei GOTO[†] Hajimu IIDA^{††} Ken'ichi MATSUMOTO[†] and Yasushi TANAKA^{†‡}

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

Kansai Science City, 630-0192 Japan

^{††} Information Technology Center, Nara Institute of Science and Technology

Kansai Science City, Nara, 630-0192 Japan

[‡] Sony Corporation, Network & Software Technology Center Software Process Solutions Department

Kounan 2-15-3, Minato-ku, Tokyo, 108-6201 Japan

E-mail: [†] {tepei-g, matumoto}@is.aist-nara.ac.jp, ^{††} iida@itc.aist-nara.ac.jp, [‡] Yasushi.Tanaka@jp.sony.com

Abstract In this paper, we propose the process model which supports reuse of a software process by using the component-oriented process model. We also present the design of the process-centered environment for project process description and execution based on our process model. The model and the system we propose are effectively supports construction and reuse of the organization's process assets, which is quite an important issue in the process improvement activities using frameworks such as CMM or CMMI level3.

Keyword CMM, CMMI, Software Process, Process Modeling, Process Centered Environment

1. はじめに

今日、ソフトウェアシステムに対する品質向上の要求から、CMM[4]や CMMI[1], ISO9001 などに基づいたソフトウェアプロセス改善活動が、開発現場で広がっている。

CMM/CMMI 段階表現のレベル3 (定義されたプロセス) にあたるプロセス改善を円滑に行うには、現在の開発プロセスを記述し、プロセス活動を分析して組織において共通の要素や性質を抽出し、それらを組織のプロセス資産として再利用することが求められる。

プロセス資産を有効に再利用するためには、それらが再利用しやすい形で形式化されていることが重要であるが、この要求を満たすようなプロセスモデリング言語として広く普及し標準化されているものは存在せず、多くの開発現場ではワードプロセッサの文書ファイルやHTML形式のWWW文書などの形でプロセス資産が記述されている。しかしながら、このような形態のプロセス資産は反復による単純な再利用はできても、プロジェクトをとりまく状況が大きく変化したときに利用できなかつたり、開発対象のプロダクトが異なると

再利用が困難であったりするため、プロセス資産の再利用が進まない原因となっている。

プロセス資産の再利用が進まないもうひとつの原因として、プロセス中心型環境[2]の不備が挙げられる。プロセス資産を再利用してプロジェクトのプロセスを定義しても、それを元に開発を進める環境が整っていないければ、定義したプロセスを活用してプロジェクト管理を行えない。そのため、プロセスを定義する工数が無駄にかかるだけという結果に陥る。

そこで我々はプロセス資産の再利用を進めるため、プロセス資産の再利用を支援するコンポーネント指向プロセスモデルと、そのプロセスモデルに基づく Web ベースのプロセス中心型環境の構築を目指して研究を行っている。本報告では、これらについて説明を行い、提案プロセスモデルによってプロセス資産の再利用が容易になることを示す。

以降 2 節では、プロセス改善モデルの一つである CMM/CMMI とそこで求められるプロセス資産の再利用について述べる。3 節では、コンポーネント指向の提案プロセスモデルについて述べる。4 節では、3 節で提案したコンポーネント指向プロセスモデルによるプロセス資産の XML 記述について述べ、5 節ではプロセス中心型環境の設計について説明する。最後に考察、関連研究の紹介とまとめを行う。

2. CMM/CMMI におけるプロセス資産の再利用

組織のソフトウェアプロセス改善において CMM (Capability Maturity Model) にあげられる能力成熟度モデルが参照される。CMM では開発組織の成熟度を 5 段階で定義し、それぞれのレベルを達成するために満たす必要があるゴールと指針が記述されている。CMMI(Capability Maturity Model Integration)は CMM を拡張・再統合したものであり、段階表現と連続表現の二つの視点から組織の開発プロセスを評価する。CMMI の段階表現では CMM 同様に組織の開発プロセスを 5 段階で評価する。CMM/CMMI 段階表現のレベル 3 では、以下に示す組織のプロセス資産を記述し、それらを再利用することが求められる。

- 組織の標準ソフトウェアプロセス
 - プロセス要素
 - プロセスアーキテクチャ
- 組織の標準ソフトウェアプロセスをテラリングするガイドライン
- ソフトウェアライフサイクル
- ソフトウェア関連文書ライブラリ
- 組織のソフトウェアプロセスデータベース

組織のプロセス資産は、SEPG (Software Engineering Process Group) と呼ばれるグループが定義する。プロ

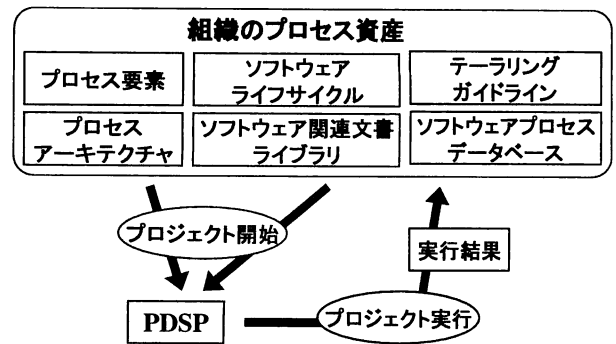


図1 プロセス資産の再利用

セス要素は開発プロセスにおけるひとつのタスクが記述されたものであり、要求分析やコーディングといった作業があてはまる。プロセスアーキテクチャ/ライフサイクルはプロセス要素間の関連を記述したもので、プロセスの構造が定義される。ソフトウェア関連文書ライブラリには過去プロジェクトで作成されたプロセスなどが保存され、ソフトウェアプロセスデータベースには実行されたプロセスに関する測定データなどが保存される。

CMM/CMMI で求められる組織のプロセス資産の再利用は、図1のように PDSP (Project's Defined Software Process: プロジェクトの定義されたプロセス) を中心に行われる。PDSP はプロジェクトごとのソフトウェアプロセスであり、組織のプロセス資産を利用して作成される。組織のプロセス資産から作られた PDSP を基に個々のプロジェクトを管理することで、プロジェクトごとの開発プロセス品質の差異を少なくすることが期待される。

PDSP はプロジェクトリーダーによってプロジェクト開始時に作成される。PDSP の作成は、たとえば以下のような手順で行なわれる。

1. プロジェクトに適したプロセスアーキテクチャを選択する
2. プロジェクトに適したライフサイクルを選択する
3. プロセスアーキテクチャとライフサイクルにプロセス要素を組み込む

PDSP は新規プロジェクトごとにプロジェクトの特徴（規模、予算、開発するプロダクト etc.）を考慮して作成しなければならず、できあがった PDSP は参照しやすい形で保存されている必要がある。

3. コンポーネント指向プロセスモデルの提案

本研究の目的は、新規プロジェクトごとに行われる PDSP の作成時に、プロセス資産を容易に再利用可能とする枠組の提供である。そのため、プロセス要素をコンポーネント化し、フレームワーク化したプロセスアーキテクチャへと容易に組み込むことができるようなコンポーネント指向プロセスモデルを提案する。

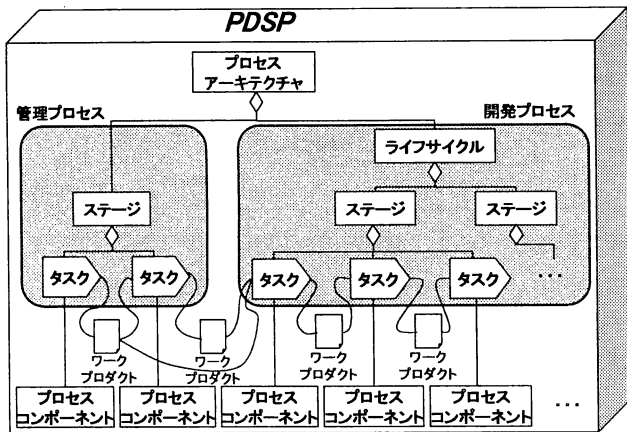


図2 PDSPの構造

3.1. PDSP 作成に対する要求

PDSPの作成に対する要求には、以下のものがある。

- 組織のプロセス資産を利用して PDSP を容易に作成できる
- プロジェクトの特徴にあわせて PDSP の変更が容易に行える
- 新規 PDSP 作成の際に過去のプロジェクトの PDSP を容易に再利用または参考にできる

これらの要求を満たすために、ソフトウェア部品のコンポーネント化とコンポーネントを組み込むフレームワークの仕組みを参考にして、プロセスモデルにコンポーネント指向の考えを組み入れる。

3.2. PDSP の構造

提案プロセスモデルでは CMM/CMMI における組織のプロセス資産にコンポーネント指向の考えを取り入れ、プロセス要素を外部とのインタフェース部分が記述されたタスクと、内部で行われる実際の作業が記述されたプロセスコンポーネントに分ける。

PDSP はプロセスアーキテクチャ、ライフサイクル、ステージ、タスク、プロセスコンポーネント、ワークプロダクトから構成される (図2)。それぞれの関係を以下に説明する。

- **プロセスアーキテクチャ**
プロセスアーキテクチャには、プロジェクト管理や要件管理など、管理プロセスのタスクと構成が記述される¹。また、システム開発のタスクをライフサイクルとして保持する。
- **ライフサイクル**
ライフサイクルには、要件定義やコーディングなど、開発プロセスのタスクと構成が記述される。ライフサイクルは開発するシステムや予算などのプロジェクトの特徴に左右されるため、プロジェ

表1 ワークプロダクト分類

分類(Group)	ワークプロダクト区分	例
Plan Group	Project Plan	個々のプロジェクトの計画
Plan Group	Estimation	見積もり
Plan Group	Organizational Plan	組織のSPI活動にかかわる計画
Engineering Group	Requirement	要件
Engineering Group	Specification	Software Productの仕様
Engineering Group	Software Product	プログラム、ソースコード、製品
Measurement Group	Measurement Data	測定データ
Measurement Group	Risk	リスク
Report Group	Record	記録
Report Group	Report	報告書
Report Group	Analysis	分析結果、アセスメント結果
Report Group	Proposal	提案、勧告
Procedure Group	Procedure	手順、是正処置、テンプレート
Measure Group	Measure	尺度
Measure Group	Criterion	基準
Resource Group	Personnel	人員、スキル
Resource Group	Tool	ツール、環境、データベース、PC
Other Group	Other	その他

クトごとに適切なライフサイクルを設定する必要がある。ライフサイクルの例としては、スパイラルやウォーターフォールがある。

- **ステージ**
ステージは、ひとつの目的を持ったタスクの集合である。ステージには、プロジェクト管理や、要求管理などがある。
- **タスク**
タスクは、ひとつのプロセス活動のインタフェースを表したものである。つまり、タスクはそのタスクに対する入出力ワークプロダクトの種類との関連を保持している。タスクは、そのタスクで行う活動の詳細が記述されたプロセスコンポーネントと結びつくことで、実際のプロセス活動を実行することができる。
- **プロセスコンポーネント**
プロセスコンポーネントは、あるタスクにおける具体的な活動内容の記述をコンポーネント化したものである。プロセスコンポーネントを対応するタスクに組み込むことで、PDSPを作成できる。
- **ワークプロダクト**
ワークプロダクトは、プロセスコンポーネントが担う各タスクで使用・作成され、タスク間でやり取りされる作業成果物である。

以降では各要素の定義について詳細に説明する²。

3.3. ワークプロダクトの分類

提案プロセスモデルでは、CMMI で言及されるドキュメントの分類を基に基本的なワークプロダクトの種類を表1のように分類した。このように抽象化されたワークプロダクトのどれを入出力とするかで、ワークプロダクトを扱うタスクを抽象化し、区分することができる。

¹ CMM の定義と厳密には異なる。

² モデルの定義はワークプロダクト分類に基づいてボトムアップに行われるので、3.2とは逆の順番で説明する。

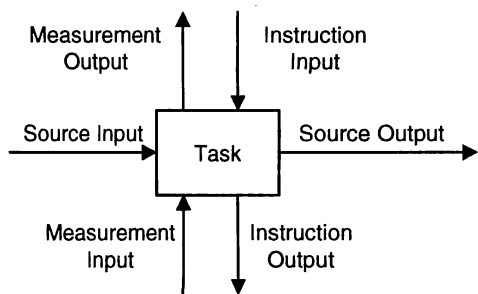


図3 タスクの入出力

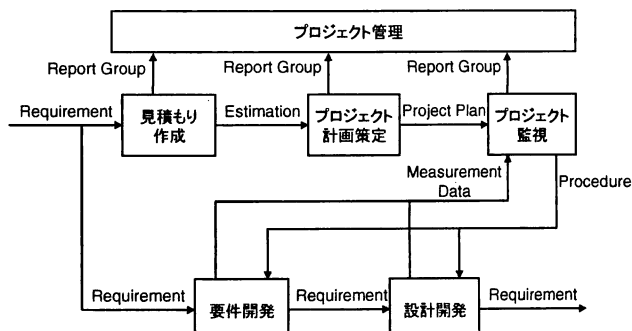


図4 プロセスモデルの例

3.4. タスク/ステージの分類

タスクは、タスクに対する入出力ワークプロダクトによって分類される。タスクの入出力は、図3のように3種の入力/出力の組に規定する。

- **Source Input/Output**
タスクが扱う主ワークプロダクト。タスクはこのワークプロダクトを変更または、作成することが主な作業となる。
- **Measurement Input/Output**
レポートや計測データなど、タスクの作業によって作成された副ワークプロダクト。
- **Instruction Input/Output**
タスクの作業を行う際の手順、計測基準を規定した副ワークプロダクト。品質目標などがあてはまる。

タスクは、以上の入出力に割り当てられたワークプロダクト型の組み合わせで分類される。例えば、「見積もり作成」のタスクは Source Input として Requirement 型のワークプロダクトを取り、Source Output として Estimation 型のワークプロダクトを出力し、Measurement Output として Report Group 型のワークプロダクトを出力する。見積もり作成のタスクを含むプロセスモデルの例の一部を図示すると図4のようになる。

ステージは、あるひとつの主ワークプロダクトを作成・管理するタスクをひとまとめにしたものである。例えば、Project Plan を主ワークプロダクトとして扱うタスクは、「プロジェクト計画」ステージのタスクとなる。

3.5. プロセスコンポーネントの構造

本プロセスモデルにおけるプロセスコンポーネントとは以下の要件を満たすものである。

- プロセスコンポーネントは、あるひとつのタスクを実現する。
- プロセスコンポーネントはそのプロセスコンポーネントが実現するタスクに定義された入出力ワークプロダクトを持つ。
- タスクで行われる活動や、責任者および作業者のロール（役割）が定義される。

プロセスコンポーネントをライフサイクルやプロセスアーキテクチャに組み込むことによって PDSP が実装される。プロセスコンポーネントを組み込む際には、その入出力ワークプロダクトの型が、組み込み先のタスクのものと一致している必要がある。

プロセスコンポーネントには、「システム要求分析」と「変更要求分析」といったように、同じタスク（要求分析）を担うものが複数存在する。同じタスクを担うプロセスコンポーネントを組み替えることで、プロセスの変更に対して柔軟かつ容易に対応することが可能である。

3.6. プロセスアーキテクチャ/ライフサイクルの構造

プロセスコンポーネントを組み込むためのフレームワークがプロセスアーキテクチャとライフサイクルである。これらには、そこで行われるプロセス活動がタスクとタスクの関連として記述される。

ライフサイクルには、プロジェクトの開始から終了までを通じて、ソフトウェアプロダクトを作成することに直接関わるタスクとその順序関係が記述される。

プロセスアーキテクチャには、ライフサイクルに記述される以外の管理プロセスのタスク（プロジェクト管理タスク、プロセス管理タスク）が記述される。

プロジェクトの特性によって左右されるライフサイクルのタスクと、それ以外の比較的可変の少ないプロセスアーキテクチャのタスクに分割することで、プロセスの再利用が容易になる。

4. 提案プロセスモデルによるプロセス資産の記述

本研究ではプロセスの記述に XML を用いる。3節で述べたように、提案プロセスモデルではプロセスアーキテクチャ/ライフサイクルにプロセスコンポーネントを組み込むことで PDSP を作成する。ここでは、プロセスアーキテクチャ、プロセスコンポーネント、PDSP の XML 表記についてデータを抜粋して説明する。

● プロセスアーキテクチャ

プロセスアーキテクチャの記述はタスク (task) とタスク間の関連 (task-relation)、タスクをまとめたス

```

<?xml version="1.0" encoding="Shift_JIS"?>
<process-architecture name="CMMIの開発プロセス" id="pa0001">
  <stage name="プロジェクト計画策定" id="scmmi0001" s-type="Planning">
    <task name="見積もりを確立する" id="tcmmi0001" t-type="CMMI:見積り作成" />
    <task name="プロジェクト計画を策定する" id="tcmmi0002" t-type="CMMI:プロジェクト計画作成" />
    <task name="計画に対するコミットメント取得する" id="tcmmi0003" t-type="CMMI:コミットメント" />
    <task-relation>
      <occurrence>
        <ordered>
          <task-ref id="tcmmi0001">見積もりを確立する</task-ref>
          <ordered exception-rule="previous">
            <task-ref id="tcmmi0002">プロジェクト計画を策定する</task-ref>
            <task-ref id="tcmmi0003">計画に対するコミットメントを取得する</task-ref>
          </ordered>
        </ordered>
      </occurrence>
      <flow>
        <wp-flow>
          <wp-type-ref>Requirement</wp-type-ref>
          <from id="stage"><stage/></from>
          <to id="tcmmi0001">見積もりを確立する</to>
        </wp-flow>
        <wp-flow>
          <wp-type-ref>Estimation</wp-type-ref>
          <from id="tcmmi0001">見積もりを確立する</from>
          <to id="tcmmi0002">プロジェクト計画を策定する</to>
        </wp-flow>
        <wp-flow>
          <wp-type-ref>Project Plan</wp-type-ref>
          <from id="tcmmi0002">プロジェクト計画を策定する</from>
          <to id="tcmmi0003">計画に対するコミットメントを取得する</to>
        </wp-flow>
      </flow>
    </task-relation>
  </stage>
</process-architecture>

```

図5 プロセスアーキテクチャのXML記述

```

<?xml version="1.0" encoding="Shift_JIS"?>
<process-component name="見積もりを確立する" id="pc0001">
  <implement-task id="tcmmi0001" t-type="CMMI:見積り確立" />
  <responsible-role r-type="ProjectManager">プロジェクトマネージャ</responsible-role>
  <work-role r-type="SoftwareEngineeringMember">ソフトウェアエンジニアリングメンバ</work-role>
  <source-input id="wp00301" status="made">製品要件</source-input>
  <source-input id="wp00296" status="made">顧客要件</source-input>
  <source-output id="wp0045" status="made">タスク記述</source-output>
  <source-output id="wp0046" status="made">ワークパッケージ記述</source-output>
  <measure-output id="wp0043">見積もりの論理的根拠</measure-output>
  <activity id="a0001">
    <guide>プロジェクトの範囲を見積もる</guide>
  </activity>
  <activity id="a0002">
    <guide>作業成果物とタスク属性の見積もりを確立する</guide>
  </activity>
  <activity id="a0003">
    <guide>プロジェクトライフサイクルを定義する</guide>
  </activity>
  <activity-relation>
    <ordered>
      <activity id="a0001" />
      <activity id="a0002" />
      <activity id="a0003" />
    </ordered>
  </activity-relation>
</process-component>

```

図6 プロセスコンポーネントのXML記述

ステージ (stage) とステージ間の関連 (stage-relation) を記述したブロックから成り立っている。これらの記述例を図5に示す。タスク間・ステージ間の関連はそれぞれの発生順序 (occurrence) とワークプロダクトフロー (flow) によって記述される。発生順序には、unordered, ordered, parallel, iterationがある。ワークプロダクトフローは、やり取りされる作業成果物の種類 (wp-type-ref) と、フロー元/先タスク (from/to) の記述でなる。

● プロセスコンポーネント

プロセスコンポーネントは、図6のようにそのプロセスコンポーネントが実装するタスク (implement-task), 入出力ワークプロダクト (*-input/output), 内部で行われる作業 (activity) とその関連 (activity-relation) からなる。

```

<?xml version="1.0" encoding="Shift_JIS"?>
<pdsp name="携帯電話用ソフト開発" id="pdsp0001">
  <project-name>A0231携帯電話用ソフト開発</project-name>
  <start date="2002-10-01" />
  <stage name="プロジェクト計画策定" id="scmmi0001" s-type="Planning">
    <pc-instance name="見積もりを確立する" id="pci0001">
      <implement-task id="tcmmi0001" t-type="CMMI:見積り確立" />
      <responsible-role r-type="ProjectManager">プロジェクトマネージャ</responsible-role>
      <work-role r-type="SoftwareEngineeringMember">ソフトウェアエンジニアリングメンバ</work-role>
      <source-input id="wpi0001" status="made">製品要件</source-input>
      <source-input id="wpi0002" status="made">顧客要件</source-input>
      <activity id="a0001">
        <guide>プロジェクトの範囲を見積もる</guide>
      </activity>
    </pc-instance>
    <pc-instance name="プロジェクト計画を策定する" id="pci0002">
      <implement-task id="tcmmi0002" t-type="CMMI:プロジェクト計画作成" />
      ...
    </pc-instance>
    <pci-relation>
      <occurrence>
        <ordered>
          <pci-ref id="pci0001">見積もりを確立する</pci-ref>
          <ordered exception-rule="previous">
            <pci-ref id="pci0002">プロジェクト計画を策定する</pci-ref>
            <pci-ref id="pci0003">計画に対するコミットメントを取得する</pci-ref>
          </ordered>
        </ordered>
      </occurrence>
      <flow>
        <wp-flow>
          <wpi-ref id="wpi0001">製品要件</wpi-ref>
          <wpi-ref id="wpi0002">顧客要件</wpi-ref>
          <from id="stage"><stage/></from>
          <to id="pci0001">見積もりを確立する</to>
        </wp-flow>
        ...
      </flow>
    </pci-relation>
  </stage>
  <wp-instance id="wpi0001">
    <implement-work-product id="wp00301">製品要件</implement-work-product>
    <file link="http://www.***.co.jp/penpal/files/010020.xls" />
    <meta-wp-data link="http://www.***.co.jp/penpal/files/010020.xml" />
  </wp-instance>
</pdsp>

```

図7 PDSPのXML記述

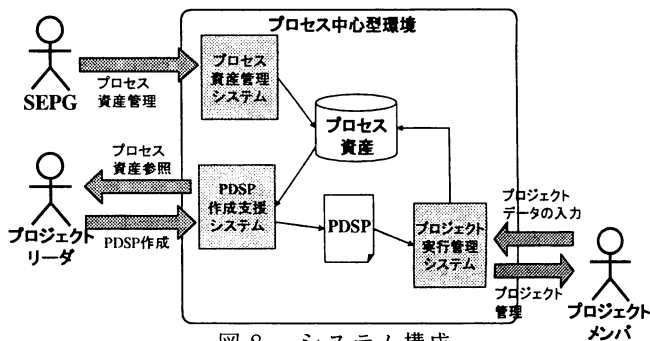


図8 システム構成

● PDSP

プロセスアーキテクチャにプロセスコンポーネントを組み込むことで、図7のようにPDSPは作成される。PDSPは、実装されたプロセスコンポーネントの記述 (pc-instance), その関連 (pci-relation) と、実装された作業成果物の記述 (wp-instance) からなる。作業成果物の記述には、その作業成果物のファイルへの参照 (file) や管理のためのメタデータへの参照 (meta-wp-data) が含まれる。

5. プロセス中心型環境の試作

提案プロセスモデルをもとに開発中であるプロセス中心型開発環境のアーキテクチャについて説明する。プロセス中心型開発環境は、Java Servletを用いて実装を行っている。プロセス中心型開発環境は、図8のよ

うに大きく分けて以下の3つのシステムで構成される。

● プロセス資産管理システム

プロセス資産は、4節で述べたように、XMLデータとして記録されている。プロセス資産の更新・追加など SEPG が行う作業を支援するのが、プロセス資産管理システムである。

● PDSP 作成支援システム

現在試作を進めているのは PDSP 作成支援システムの部分である。PDSP の作成は、2節で示した手順でプロジェクトリーダーによって行われるが、各ステップでは以下のような支援がシステムによって行われる。

1. プロセスアーキテクチャの選択：

プロジェクトの規模、スケジュール、目的などのプロジェクトの特徴を元に、過去の似たようなプロジェクトで用いられたプロセスアーキテクチャを提示する。プロジェクトリーダーは、その中から適切なものを選択することができる。

2. ライフサイクルの選択：

プロセスアーキテクチャと同様にして、適切なライフサイクルの候補を提示する。また、プロジェクトリーダーはスパイラルのイテレーション回数を決めるなど、ライフサイクルのテーラリングを行う。

3. プロセスコンポーネントの組み込み：

タスクのインタフェースに合致するプロセスコンポーネントをシステムが提示し、プロジェクトリーダーはその中から適切なものを選択する。

● プロジェクト実行管理システム

プロジェクト実行管理システムでは、完成した PDSP を元にプロジェクトの実行支援を行う。ファイル管理や、インスタントメッセージャーによるプロジェクトメンバーのコミュニケーション支援などを行う予定である。

6. 考察

6.1. 提案モデルの定性的評価

提案プロセスモデルでは、プロセスアーキテクチャ／ライフサイクルにプロセスコンポーネントを組み込むことで容易に PDSP を作成することができる。また、プロジェクトの特徴にあわせて PDSP を変更する際にも、現在用いているプロセスコンポーネントの一部を別のプロセスコンポーネントに入れ替えることで、多くの場合に対応可能である。

過去の PDSP の参照についても、PDSP が構造化され、それぞれのタスクがコンポーネント化されているため、必要な部分を抜き出して参照することが容易である。

6.2. 関連研究との比較

これまで、様々なプロセスモデリング言語とプロセ

ス中心型環境が提案されてきた[2]。また、Gary らによる OPC (Open Process Component) [3]のように、コンポーネント指向の考えを取り入れたプロセスモデルも提案されている。

一方で、UML によってソフトウェアプロセスを記述する試みが OMG で行われている (SPEM: Software Process Engineering Metamodel) [5]。これは UML の拡張機能を用いてソフトウェアプロセス要素の記述方法を定義しようというもので、2002年に OMG にドラフトが提案されている。

このように、様々なプロセスモデリング言語とプロセス中心型開発環境の研究が行われているが、どの研究の成果も広く世間で使われるには至っていない。また、CMMI などのプロセス改善の要求が高まっているのに対して、ソフトウェアプロセス資産の再利用に焦点を当てたものは少なく、XML などの標準を利用する試みも十分行われているとは言い難い。

我々の研究では、プロセス資産の再利用を主たる目的として、コンポーネント指向や XML を取り入れ、プロセス改善を支援できるプロセスモデルの構築を目指している。

7. まとめと今後の課題

本報告では、CMM/CMMI で規定されているプロセス資産の再利用に適したプロセスモデルの提案と、そのプロセスモデルを用いてプロセス資産の再利用を支援するプロセス中心型開発環境の設計を示した。現在、このアーキテクチャに基づくプロセス中心型開発環境の試作を行っているところである。

現在のところ我々のプロセスモデルは XML によるテキスト的な表現しか持っていないが、プロセスモデルにはグラフィカルな表現も必要であると考えている。そこで SPEM の成果を参考に、本プロセスモデルのグラフィカルな表記を定義することも今後の課題として挙げられる。

文 献

- [1] CMMI Product Team, Capability Maturity Model Integration Version 1.1 Staged Representation, Software Engineering Institute, CMU/SEI-2002-TR-012, 2002.
- [2] 井上克郎, 松本健一, 飯田元, “プロセス記述言語と開発環境,” ソフトウェアプロセス, pp.43-96, 共立出版, 2000.
- [3] K. Gary, T. Lindquist, and H. Koehnemann, “Component-based Software Process Support,” Proc. 13th Conference on Automated Software Engineering, Honolulu, Hawaii, October, 1998.
- [4] M. C. Paulk, B. Curtis, M. B. Chrissis, and C. V. Weber, Capability Maturity Model for Software Version 1.1, Software Engineering Institute, CMU/SEI-93-TR-24, 1993.
- [5] OMG, Software Process Engineering Metamodel Specification (SPEM), An Adopted Specification of the Object Management Group, formal/02-11-14, November, 2002.