ソフトウェア工学 71－4
（１９９０．２．８）

ＪＳＰ法を用いた設計プロセスの記録と分析

中島 毅、 田村 直樹、 上原 憲二
三菱電機（株） 情報電子研究所

ＪＳＰ法を用いた実設計プロセスを記録し分析を行なった。設計プロセスの記録と分析には、我々が提案してきているＰＰＫ法(Problem-Product-Knowledge) を用いた。本手法は、記録の方法とインタビューによる記録の補充と記録を整理する枠組からなっている。

分析では、実プロセスとＪＳＰ法との間の溝を明らかにすることに重点をおき、さらに、実プロセスに現れる様々な知識を分類した。本報告では、その分析結果を報告する。

Recoding and Analyzing a JSP Design Process

Tsuyoshi NAKAJIMA, Naoki TAMURA, Kenji UEHARA
Information Systems and Electronics Development Laboratory
Mitsubishi Electric Corporation
5-1-1 Ofuna, Kamakura, Kanagawa, 247 Japan

We recorded and analyzed an actual process, which a designer designed an example with JSP method. To record and analyze the process, we use Problem-Product-Knowledge method (PPK method), which we have proposed. Our method consists of a method for recording activities, interview technique, and a framework for organizing the record.

In this paper, we turned out several problems which lay between JSP method and an actual process. Furthermore, we classified Knowledges in the PPK description.

1．はじめに

近年、ソフトウェアプロセスに関する議論が盛んであり [1] [2] [3]、多くのプロセス記述言語が提案されている。しかし多くの場合、それらの記述言語の支援対象や記述力に対する評価は曖昧である。

実プロセスを記録し分析することは、プロセスに共通する多くの有用な仮説を生み出し、一方でそれらの仮説を評価するための例題を提供してくれる効果をもっている。しかし、今までのところ数少ない例しか報告されていない。この理由は、第一に、実プロセスの記録をとる行為自体が多大な労力を要すること、第二に、膨大な実プロセス記録をまとめる統一的な枠組がないことである。

我々は、このような問題点を克服すべく、設計プロセスの記録法として、ＰＰＫ (Problem-Product-Knowledge) 法を提案してきており [4] [5]、さらにそれに基づき、プロセスの記録と直接の支援を狙ったツールを構築中である [6]。

一方、ＪＳＰ法 [7] などの設計手法は、最近支援ツールの普及もあって、現場においても徐々に脚光を浴びつつある。しかし、設計手法は、習得が難しく、必ずしも良い設計結果にならないという批判もある。設計手法と実プロセスの間にある溝の正体を明らかにしていく必要がある。

今回、ＪＳＰ法を用いて簡単な課題を解く実験を行なった。その際ＰＰＫ法によって設計プロセスを記録してもらい、その記録を種々の視点から分析した。本論文は、その分析結果について報告するものである。

2．設計プロセスの記録法

ＰＰＫ法は、設計者が記録を行ない、その記録を基に分析者が一つの実プロセスとして記述するものである。ＰＰＫ法は次の３つの部分から成っている。

(1) 記録の方法
(2) インタビューによる記録の補充
(3) プロセス記述の枠組（ＰＰＫモデル）

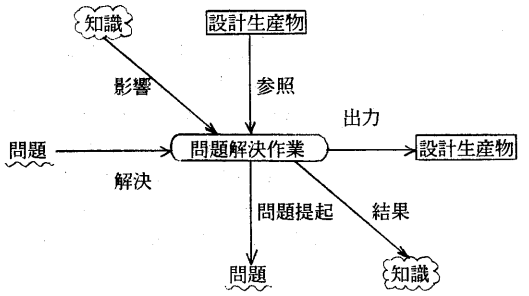
ここでは、(1)と(2)に関する説明を省き、プロセス記述の枠組のみを簡単に説明する（詳細は [5]）。

・ＰＰＫモデル

ＰＰＫ法の前半部分（記録(1)とインタビュー(2)）を行うと、問題と設計生産物と知識の三つの情報が生じる。ＰＰＫモデルは、これらの記録された情報を、「問題解決作業」（以後「作業」とする）で関連付けて連鎖を作るものである（図１）。

一つの作業は、三つの情報を入力し新たな三つの情報を出力する。三種類の出力は、それぞれ他の作業への入力となることができるので、ＰＰＫモデルで記述された実プロセスは、ネットワーク図となる。

ＰＰＫモデルは、概念的には、問題解決プロセス（問題－作業－問題）と生産物作成プロセス（生産物－作業－生産物）の二つの軸を融合させたものである。ＰＰＫモデルの特徴的なところは、この二軸で捉えきれない作業の意味づけを「知識」という枠組によって表現することである。



入力　問題　　解決すべき問題・動機
　　設計生産物　作業で参照が必要なもの
　　知識　　作業に影響を与えた頭の中の考え
　　例：・設計手法、前の設計のときの経験
　　・大まかな予想、実現イメージ
出力　問題　　作業中に生じた問題点・動機
　　設計生産物　作業で作成された生産物
　　知識　　作業で得た頭の中の考え
　　例：・実現イメージ、達成感

図１．ＰＰＫモデル

3．記録実験

3.1 実験の条件

(1) 課題

設計課題は、文献 [8] のものを用いた。この課題は、簡単ではあるが再帰的処理を含んでおり、単純にＪＳＰ法では解けないものを選んだ。さらに、仕様の曖昧さが実プロセスに与える影響を調べるために、課題の仕様を故意に曖昧した。

> 語の直接的な親子関係を示す行を次のように示す。
> parent : child1 child2 child3 ...
> ここで、':' で区切られた最左の語が親を表し、右の語群がその子どもを表す。子ども間は空白文字で区切られているものとする。この形式の行からなるファイルが与えられたとき、次の３つの解析結果を出力するプログラムを作れ。
> 課題１　指定された語の子孫となるすべての語
> 課題２　指定された語の子孫でないすべての語
> 課題３　親として定義されていない語

(2) 制約　・設計手法としてＪＳＰ法を用いる。
・ＰＰＫ法の記録手順を用いること。
・コーディングまで行なうこと。

(3) 設計者　・Ｃのプログラミング歴４年
・ＪＳＰ法を勉強中（解説書を読んだが実際の問題を解いたことはない）

3.2 結果

(1) 設計の記録・・・Ａ４の紙２４ページ

(2) プログラム・・・shellスクリプト　１００行
ロードモジュール　６個
（Ｃプログラムで５７３行）

(3) 所要時間・・・・約１６時間（３日間）

(4) ＰＰＫモデルによる記述

図４は全設計プロセスをＰＰＫモデルで記述したものである。

４．実プロセスとＪＳＰ法の対比

この章では、実プロセスの中でＪＳＰ法以外の部分を、作業の流れ・作業・生産物の三つの視野から明確にしていく。

4.1 作業の流れ

図２は、図４のＰＰＫモデルによる記述を意味的にまとめ、問題のノードを記述から取り除いたものである[4]。図２には、さらにＪＳＰ法との対応関係を右側に記した。

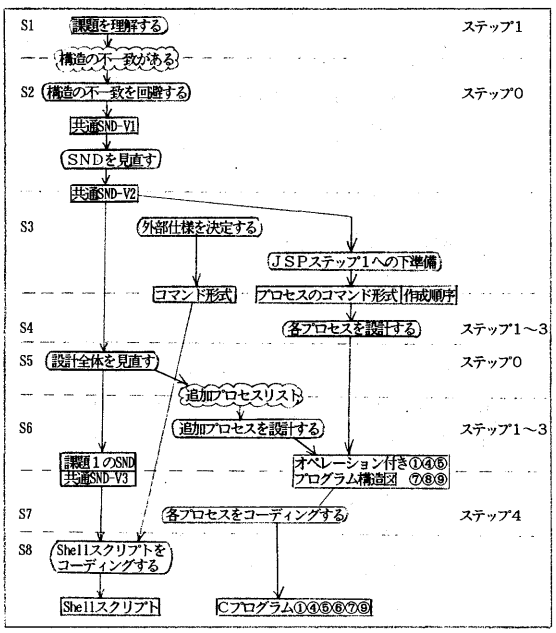


図２．実プロセスとＪＳＰ法との対応

図２から、実プロセスはＪＳＰ法の基本ステップと比較的よい対応関係ををもっていることがわかる。二箇所で手戻りが生じている。S1→S2とS4→S5である。この内、前者はＪＳＰ法に記述のある手戻りである。すなわち、「（ステップ２）入出力データの対応がとれないなら、（ステップ０）構造の不一致を回避せよ」に相当する。

これに対して、S4→S5で生じている手戻りは事情が異なる。この段階は、ＪＳＰ法のステップ３→ステップ０に相当するが、これに対する手戻りの起きる条件と戻り先に関する記述がＪＳＰ法の解説書[7]中には見当たらない。

この手戻りの直接の原因は、ステップ０で完成しているはずのシステムネットワーク図（ＳＮＤ）が不完全であったことにこの時点で気がついたことである。後のインタビューで設計者自身が述べたところによると、ＳＮＤの書き方に慣れていなかったことが原因である。

4.2 作業

ＰＰＫモデルで記述された実プロセス（図４）の「作業」を、ＪＳＰ法に指定された作業とそうでないものに分けると以下のようになる。

ＪＳＰ法が指定する作業　７０
ＪＳＰ法が指定する作業以外　３９

約１／３が、ＪＳＰ法で指定されていない作業であることが分かる。この内、最小の作業単位に相当するものを抜き出しその出現個数をカウントしたのが表１である。

表１．ＪＳＰ法で指定されていない作業

作業	出現個数
仕様を決定する	7
コマンド形式を決定する	
入力データの文法を決定する	
エラー処理を決定する	
解き方を決定する	4
表現法を決定する	
解法を決定する	
アルゴリズムを変更する	
手順を決定する	4
作業手順を決定する	
作成順序を決定する	
コーディング順序を決定する	
質問する	4
出題者へ質問する	
回りの人に質問する	
テストケース表を作成する＊	1
テストデータを作成する＊	1
テストデータでトレースする＊	1
名詞を抜き出す	1
似た例題を探す	1
結果をまとめる	1
複写する	1
修正する	1
選択する	1

これらの作業は次の３つのカテゴリに分類できる。

(1) 実設計プロセスで必要なステップ
　○仕様を決定する
　　テストをする（＊をまとめた）
(2) 手法の生成・検索
　○手順を決定する
　　解き方を決定する
　　似た例題を探す
(3) 一連の作業の中で意味をもつ具体的な活動
　　質問する
　○複写する・修正する（ＪＳＰ法ステップ３）
　　選択する
　　結果をまとめる
　　名詞を抜き出す

次に、○がついた作業について、実プロセスにどのような影響を与えているかを見ていく。

○仕様を決定する

課題は故意に仕様を曖昧にしておいた。これに対して「仕様を決定する」作業がS2～S7の割合広い範囲にわたって７回行なわれている（図４の仕様決定１～７）。作業の出現の仕方には、その緊急度によって二つのパターンがあった。

・割り込み的　　（仕様決定５７）
　気がついたその場で仕様決定が行なわれ、終了後元の作業に戻ってくる。
・作業の区切りで　　（仕様決定１２３４６）
　前の作業が一段落したところで行なわれる。

○手順を決定する

実プロセス中では四箇所で生じている（図４手順決定１～４）。これらの「手順を決定する」作業は、その時点での状態を反映し、動的に手順を生成する。以後の作業はこの手順に従って行なわれている。内容的には、次の二つの場合があった。

・作業の整理（手順決定１）
　手順決定１は、ＪＳＰステップ０が一段落しＪＳＰステップ１への準備段階に生じている。やるべきことが多く、やり残しと手順前後が不安になったので、やるべきことを整理している。
・ＪＳＰ法で曖昧な順序の決定（手順決定２３４）
　実プロセスのS4は、ＪＳＰステップ１～３に相当する。設計すべきプロセスが五つあって、その作成順序を決めるための記述はＪＳＰ法にはない。手順決定２では、「難しい順に作成する」という基準で順番を決めている。手順決定３についても同様である。

手順決定４は、ＪＳＰステップ１から２の前半を終った時点で、ステップ２の後半へ進むか、次のプロセスの設計へ進むかを決定している。

○複写する・修正する

この二つの作業は、図４の「P7の設計」で連続した作業として現れている。同じＪＳＰステップを行なうときでも、異なる実作業となって現れる一例である。

4.3 生産物

ＰＰＫモデルで記述された実プロセス（図４）の生産物を、ＪＳＰ法に指定された生産物とそうでないものに分けてその数を数えたのが表２である。

ＪＳＰ法で指定されていない生産物は、全生産物の約２割程度であり、そのすべてがＪＳＰ法で指定されていない作業（4.2 節）の出力である。また、これらの内、明確な形式をもった生産物は、テストケース表・テストデータ・トレース結果・エラー表の四つにすぎずで、残りの多くは、簡単な絵もしくはメモ書きである。

表２．生産物と出現個数

生産物	出現個数
ＪＳＰ法で指定されている生産物	66
課題	3
ベン図	3
ＳＮＤ	14
入出力データ構造図	18
プログラム構造図	7
オペレーションリスト	7
オペレーション付きプログラム構造図	7
ソースコード	7
shellスクリプト	3
ＪＳＰ法で指定されていない生産物	15
コマンド形式	5
作業手順	2
テストケース表	1
テストデータ	1
トレース結果	1
エラー表	1
表現図	1
場合分け結果	1
他の解法	1
名詞リスト	1

5．知識の分類

ＰＰＫモデルで「知識」をさらに細分化しなかった理由は、設計プロセスを構成する知識の構造が今だ明確でないことと、要素数をいたずらに増やせば、モデルの単純さが失われ、記述上の制約になると判断したからであった。

しかし、実プロセスの分析という視野に立てば、知識の分類はどうしても必要である。この章では、実プロセスの記述データを用いて「知識」の分類を行なった結果を示す。

図４から抜きだした「知識」は、次の５つのパターンに分類できる。

(1) 作業の進め方・・・現在の状況を判断して、以後の作業の進め方を決めるための情報。
(2) 発見・見通し・・・作業中に発見したアイデアや事実や見通し。後の作業に強い影響を与える。
(3) 達成感・・・（設計者にとって）作業が意味的に一段落したときに感じる主観的な達成度。
(4) 確認・・・設計結果の正しさを確認した結果。
(5) 決定事項・・・仕様や実現方式に関する決定。本来仕様書等の生産物に記述されるべき性質のものである。

この分類に従い、図４の「知識」を分類したものが図３である。

(1) (2)は「なぜそうしたのか」、(3)～(5)は「その結果どうなったか」を説明する。(1)～(5)は他人の設計プロセスを理解する際、有用な情報である。

(2)～(5)が実プロセスに付随したものであるのに対して、(1)は実プロセスから独立しており、直接再利用可能な「知識」と言える。(1)には、設計手法が示す基本ステップ、生産物の形式と書き方、例題の他に、設計者が経験的にもっているルールや、実プロセスで獲得していくルールがある。興味深いのは、（ともにインタビューで直接引き出したのであるが）経験的にもっているルールが、獲得したルールよりも抽象度が高いことである。

もちろん、経験的なルールは、図３の(1)のような形式では、設計者の高度な解釈がどうしても必要である。これらの知識を適用する際、次の二つの情報を推測しなければならないからである。

適用条件　　どんな場面で適用できるか？
具体的な活動　どのように適用するのか？

(1) 作業の進め方
・設計手法（ＪＳＰ解説書）
基本ステップ０～４
生産物の形式と書き方
手戻りの指定
構造の不一致を扱う例題
・経験的にもっているもの
前の結果に自信がなければ、不安要因を含むものから先にレビューする
先にすべてのプロセスについて見通しを立てた方がよい
小さなプログラムから作っていくとテストが容易になる
・獲得したもの
構造の不一致を持つプログラムの解はＳＮＤで表現し検討すべきだ
照合問題であり入力が分割されていない場合、入力ファイルを条件に従って中間ファイルに分割するプロセスを考える
照合問題であり入力がソートされていない場合、sortプログラムを挿入すればよい
(2) 発見・見通し
他の課題も同様のステップで解けるだろう
再帰処理はスタック的に実現できそうだ
プログラム全体の構造の概略が分かる
課題１は再帰的な処理を含むアルゴリズムだ
課題１と２は似ている
照合問題に似ている
照合問題である
二重の照合問題である
課題３が簡単そうだ
P1と似ている
P1の中に埋め込むことができる
課題２＝全体－課題１で解ける
構造の不一致が起きている
子どもを新たなキーとする処理となる
(3) 達成感
課題３はＪＳＰ法で解けた
個々のプログラムと中間ファイルが明確になる
shellプログラミングが可能と確信する
(4) 確認
構造の不一致はない
ＪＳＰステップ３まで終了
共通SND-V2に誤りはない
(5) 決定事項
・仕様
出力は一行８０文字
語は８文字認識
プログラム名が決まる
出力は一回だけでよい
課題２＝全体－課題１がよい
・実現方式
案１がよい
キーも子孫になれる
キューを用いるアルゴリズムで解こう

図３．実プロセス中に現れた全「知識」

６．おわりに

実プロセスとＪＳＰ法との対比を通じて、ＪＳＰ法がカバーし切れていない部分を明らかにし、さらに、実プロセスにおける様々な知識を分類した。これらの結果は、直接的には、設計手法を支援するツールを構築する際に有用な情報であるとともに、設計プロセスの再利用可能な記述形態（記述モデル）の研究に役立つものと信じている。

今後は、他の設計手法による設計などの記述実験を通して、ＰＰＫ法とその分析法を充実させていく予定である。

参考文献

[1] 1st-5th ISPW, ACM SigSoft
[2] L. Osterweil, "Software Processes Are Software Too," ICSE 9, IEEE, 1987, pp14-16
[3] K. Kishida and others, "SDA: A Novel Approach to Software Environment Design and Construction," ICSE 10, 1988, pp69-79
[4] 中島他，ソフトウェア設計における実プロセスの記述法，ソフトウェアシンポジウム'89, pp187-196
[5] 中島他，ＰＰＫ法：ソフトウェア設計プロセスの記録と分析の手法、ＳＥ研究会67-2(1989-7)
[6] 田村他，ハイパテキストを用いた設計プロセス支援ツールの試作、ＳＥ研究会68-(1989-9)
[7] J. Cameron, "JSP&JSD: The Jackson Approach to Software Development," IEEE Computer Society Press
[8] B. Cox, "Object Oriented Programming"（邦訳）「オブジェクト指向のプログラミング」トッパン

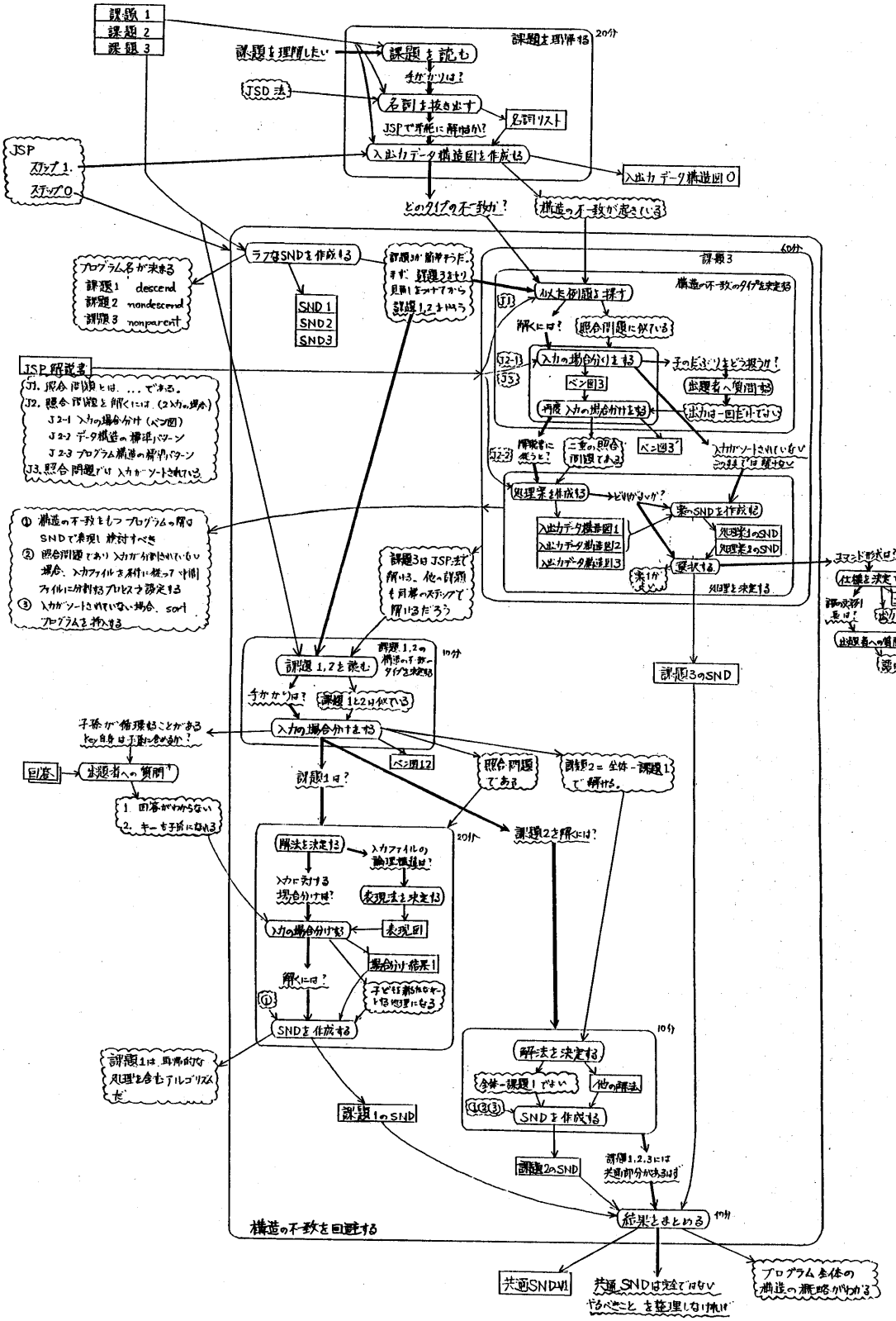


図4．ＰＰＫモデルによる全設計プロセスの記述（１／３）

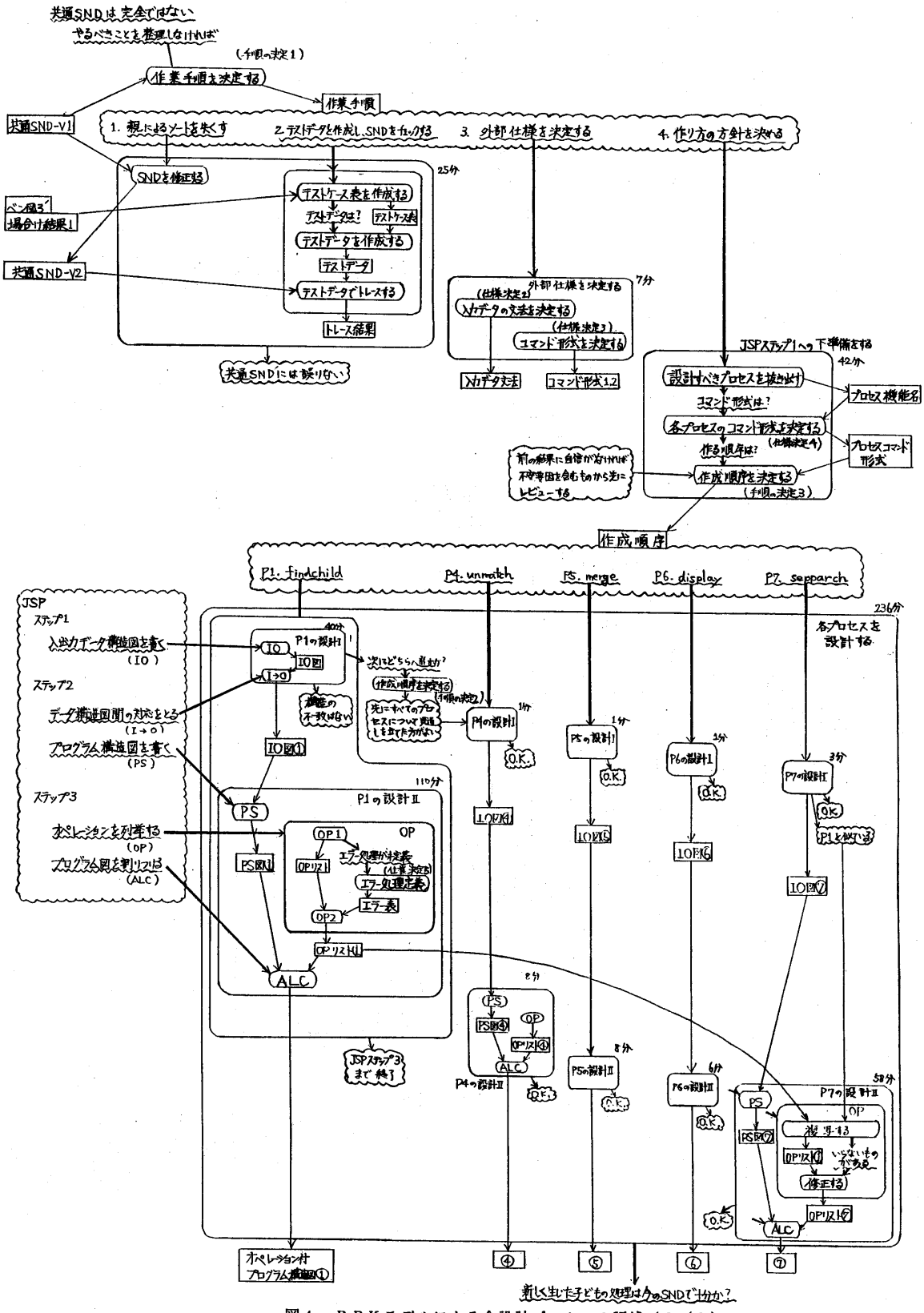


図4．ＰＰＫモデルによる全設計プロセスの記述（2／3）

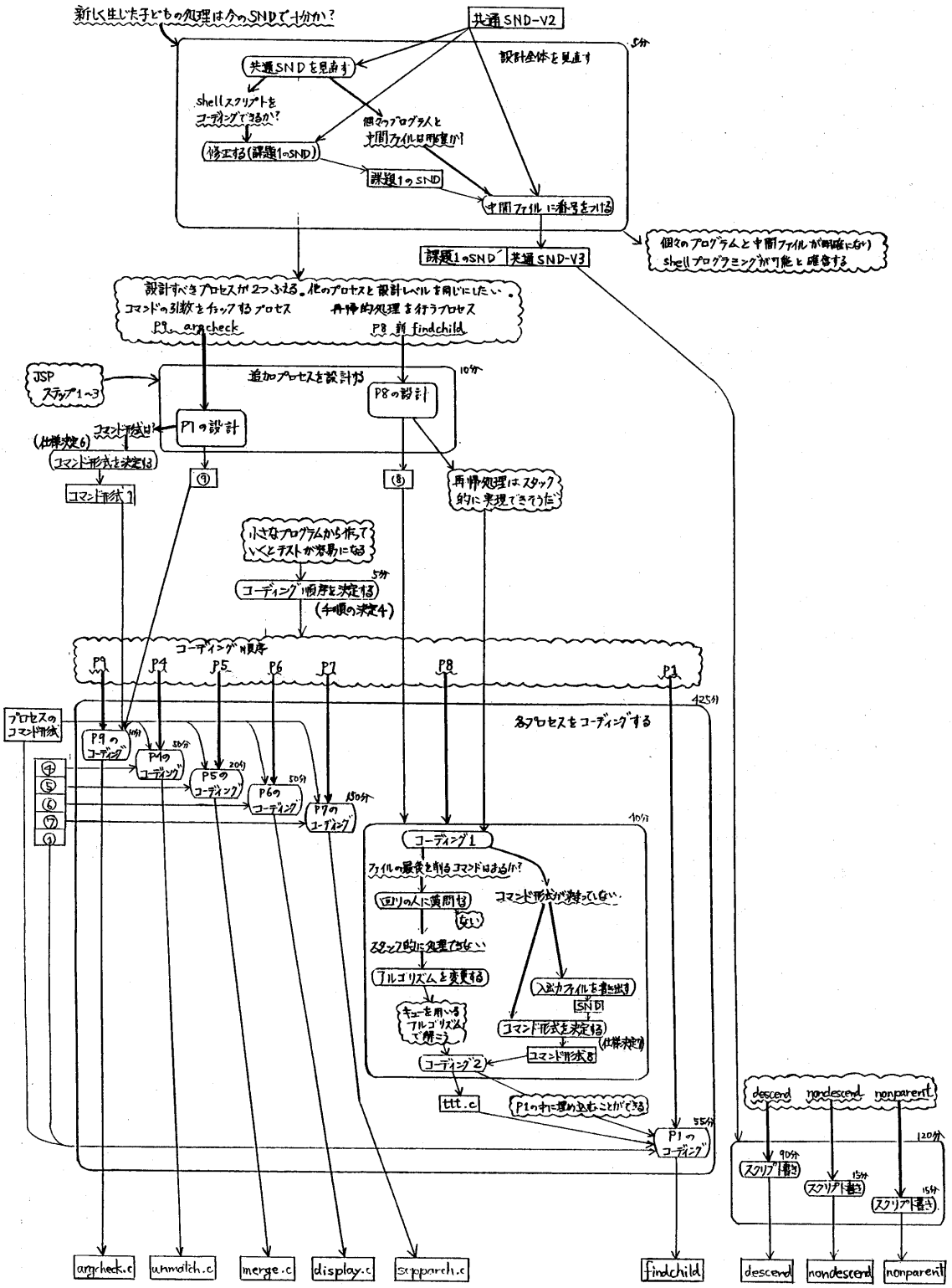


図4．ＰＰＫモデルによる全設計プロセスの記述（３／３）