ソフトウェア工学 67－2
（１９８９．７．２５）

# ＰＰＫ法：ソフトウェア設計プロセスの記録と分析の手法

中島　毅、　田村　直樹、　藤岡　卓、　上原　憲二、　高野　彰
三菱電機（株）　情報電子研究所

本報告では、ソフトウェア設計プロセスを記録し分析するための手法を提案する。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る。この枠組で記述されたプロセスは、二つの視点から眺めたり組織化したりできるので、理解しやすく分析しやすい性質を持っている。

本報告では、本手法を紹介し、分析例として実プロセスの記録から設計手法の雛型を導出することを議論する。

## PPK: A Method For Recording And Analyzing Actual design Processes

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

We have developed a method for recording and analyzing actual design processes, which consists of a method for recording activities, interview technique, and a framework for organizing the record. The framework allows organizing a record into a process description and provides two views of the description. This helps analyzing actual design processes.

In this paper, we explain our method and discuss how to extract a design method from actual processes recorded by using the method.

## 1．はじめに

近年、ソフトウェアプロセスに関する研究が盛んである[1]。種々のプロセスモデルが提案され、作業をプログラミングする研究もなされている[2]。設計段階の支援が重要視される中で、設計プロセスのモデル化と支援に関する議論も起こってきている[3]。しかし、設計プロセスは人間的要因ゆえに複雑で捉えどころがない。こうした中で実設計プロセスの記録と分析の必要性が増してきている[4][5]。

本報告では、設計プロセスを記録し分析するための設計プロセス記述法を提案する。この記述法では、設計プロセスを、小さな問題・生産物・知識の連鎖として記録していく。この記述法の大きな特徴は、その記録を種々のビューで見たり、構造化したりできることであり、設計の支援自体よりも分析に重点をおいている[6]。本手法で記述された設計プロセスを分析することによって、設計の手順と知識を抽象化し、それを反映した良好な設計プロセスを開発することを狙っている。

本報告では、第2章で記述法を紹介し、第3章で記述法の実際を例を用いて示していく。第4章では実設計プロセスから設計手法を導くための手順について考察する。

## 2．設計プロセス記述法の提案

### 2.1 設計記述法の要件

実設計プロセスを記述することを考えると、記録を行う設計者と、記録をまとめ分析を行う分析者の立場を考慮する必要がある。この場合、設計プロセスの記録と記述のための方法として求められる事項は次の四点である。

(1) 設計者の設計作業をなるべく妨げないこと。
(2) 結果としての記録には、後で分析するに十分な情報を含むこと。特に設計の際の思考の流れをトレースできることが重要である。
(3) 分析者が、記録を基に容易に記述できること。
(4) 結果としてのプロセス記述は、わかりやすく読みやすいこと。さらに分析のため、いくつかのビューでみせたり、組織化できる記法であること。

我々が提案する設計プロセス記述法は、上記の条件を満たすことを狙ったものであり、次の三つ部分からなる。

(1) 記録の方法 → 2.2
(2) インタビューによる記録の補充 → 2.3
(3) プロセス記述の枠組 → 2.4

### 2.2 記録の方法

設計者は以下の手順に従って記録をとっていく。

設計が終了したと思うところまで(1)-(3)を繰り返す。
(1) 解決すべき問題を挙げる(a)。開始時間を書く。作業番号を付ける。
(2) 絵等(b')を書きながら、自由にその問題を解決していく。そのとき、次のことに注意する。
　①考えていたことを網羅的に書き出す(c')。
　②参照していたもの（前の作業で書いた絵ならその作業番号）を書き留める(b)。
　③思いついた問題点を書き出す(a')。
　④中断／再開があったら、その時刻とその理由を書き留める。
(3) (2)の作業が一区切りついたらこれまででわかったことや達成感を記す(c')。終了時間を書く。

図1．記録手順

結果として、この記録方法では、時間情報と、三つの種類のデータ、問題（a, a'）と設計生産物（b, b'）と知識（c, c'）、の記録を行う。

### 2.3 インタビューによる記録の補充

分析者はインタビューによって記録を補充する。インタビュー手順は、以下の通りである。

(1) 記録を読み、理解できないところに印をつける。
(2) (1)で印をつけた部分と次の三点に注意して、設計者にインタビューする。
　①何のためにこの絵を描いたか？ (a)
　②絵を書いているときに何を考えていたか？ (c)
　③結果として何を得たのか？ (c')
インタビュー結果は、記録の余白に書き込んでおく。

図2．インタビュー手順

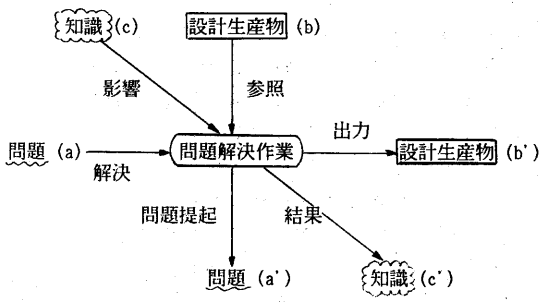
### 2.4 プロセス記述の枠組

記録とインタビューを行うと、問題と設計生産物と知識の三つの情報が生じる。これらの記録された情報は、問題を解決する作業によって関係づけることができる。

我々が提案する設計プロセスの記述の枠組は、これら三つの情報を「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り、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。その記述モデルをＰＰＫモデル（図３）（Problem-Product-Knowledge model）と呼び、プロセスを記述する方法全体(2.2-2.4)をＰＰＫ法と呼ぶことにする。

ある作業の３種類の出力は、それぞれ他の作業の入力となるので、ＰＰＫモデルで記述されたプロセスは、ネットワーク図となる。




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３．ＰＰＫモデル

## ３．ＰＰＫ法の実際

我々は、既に三つの実プロセスの記述実験を行っている。これらの実験を通じて、記述する上で気をつけるべき点がいくつか明らかになってきているので、例を用いながら示していく。

### 3.1 記録とインタビュー

実際に 2.2の手順で設計者に記録してもらい、さらに 2.3の手順でインタビューした結果を図４に示す。

設計の記録は、思考のプロセスを妨げないように、設計者になるべく自由なやり方で行ってもらう。経験的に次のデータが収集しにくいことがわかっている。

(1) 解決すべき問題 (a)
　目的意識がはっきりしない場合でも作業が開始する場合がある。
(2) わかったことや達成感 (c')
　言葉として表現しにくい。また、次の作業が始まるとき新しい問題に関心が移ってしまうので記録を怠ってしまうことが多い。
(3) 作業の背景となるような知識 (c)
　設計手法や以前の経験などを作業中に書くことは難しい。

これらの点は分析者がインタビューを行うことで補充する。この際、記録時に頭に浮かんだ考えを書き留めておいてくれていることが重要である。つまり、知識 (c, c') は、設計者に対してインタビューを行う際「何を考えていたか」を思い出すためのキーとして使うことができるからである。

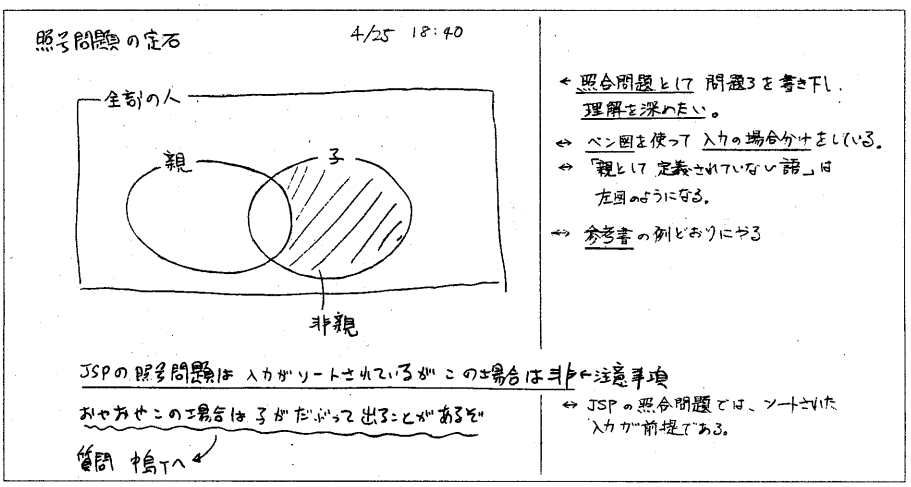



図４．設計の記録とインタビュー結果（左：記録／右：インタビュー結果）

図４の記録部分では、出力問題は記録してあるが、作業の目的ばかりか、何をしているのかもわからない。これらの情報はインタビューによって後から補充している。

しかし、参照していたもの(b)、新しい問題点(a')の発生原因、中断／再開の時刻と理由のように、インタビューによって引き出しにくい情報もある。これらは比較的書きやすい情報でもあり、予め設計者に注意してもらう方が望ましい。

### 3.2 記述の方法

記録とインタビューで得た記録をＰＰＫモデルで記述する際、次の点に注意する必要がある。

(1) 適切な名前をつける

記述で一番注意を要するのは、作業と生産物に適切で短い名前をつけることと、問題と知識を簡潔にまとめることである。一般的に、人の作業を理解する枠組として、行動の型・状況・主体・対象・手段方法・目的・結果がある[7]が、これらはＰＰＫモデルとは次のような対応関係をもっている。

行動の型 → 作業
主体 → なし
対象 → 入出力生産物
目的 → 入力問題
結果 → 出力問題、出力知識
状況・手段方法 → 入力知識

これらを記述の中で適切に表現していくことがわかりやすいプロセス記述を書く上で大切である。

図４の記録に基づいて、ＰＰＫモデルで設計プロセスを描いた結果が図５である。記録の中から、作業名と出力生産物名、入力問題を決めている。

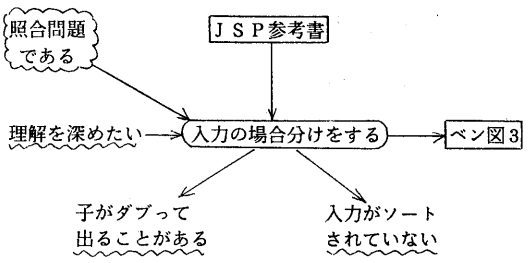



図５．図４の記述からＰＰＫモデルで記述する

(2) 作業を意味的にまとめる

記録を単に一面に広がったネットワーク図にしたのでは、作業の大きな流れを摑みにくい。ＰＰＫモデルの利点の一つは、作業を組織化できることである。これによって、いくつかの作業と三種類の情報を一つの丸で囲んで、より大きな作業にまとめることができる。これは、設計プロセスのマクロな流れを見つける上で重要である。

プロセス記述のどこを丸で囲むかについては、次の点に注意すべきである。

①適切な作業名をつけることができる。

簡潔な名前で、意味的にまとまめられることが第一の条件になる。この際、出力知識に注意する。作業後の達成感は、直前の作業に関するものだけでなく、一連の作業の結果として現れていることが多い。これが作業をまとめるためのキーとなる。

②多くの生産物と知識を内部に隠すことができる。

③入出力の問題と生産物と知識を少くできる。

作業を意味的に作業をまとめることができれば、設計プロセスは格段に理解しやすくなる（図６）。

### 3.3 例題と記述

本手法を用いて実設計プロセスを記録した。この設計で取り扱った例題と設計者の経験等ついて次に示す。

例題 ： 個人用簡易データベース
設計の中心・・・簡易問合せ言語の設計
と表操作ルーチン
規模 ・・・Ｃソース約２Ｋ行
仕様 ・・・あいまい
経験 ： Ｃプログラミング歴４年
YACCの使用経験あり
言語設計経験なし

この設計プロセスの前半部分を記述した結果を図６に示す。図６は、問題発生のプロセスとしての視野を提供している。この視野は設計者の思考の流れを追いやすくする[6]。

## ４．実設計プロセスの分析と設計手法の導出

この章では、3.3のプロセス記述（図６）を例にとり、実プロセスの悪い部分を修正し、さらに設計手法を導く方法について議論する。4.1と4.2では、設計プロセスを良くする方法、4.3ではその結果に基づいて設計手法を抽出する方法を議論する。

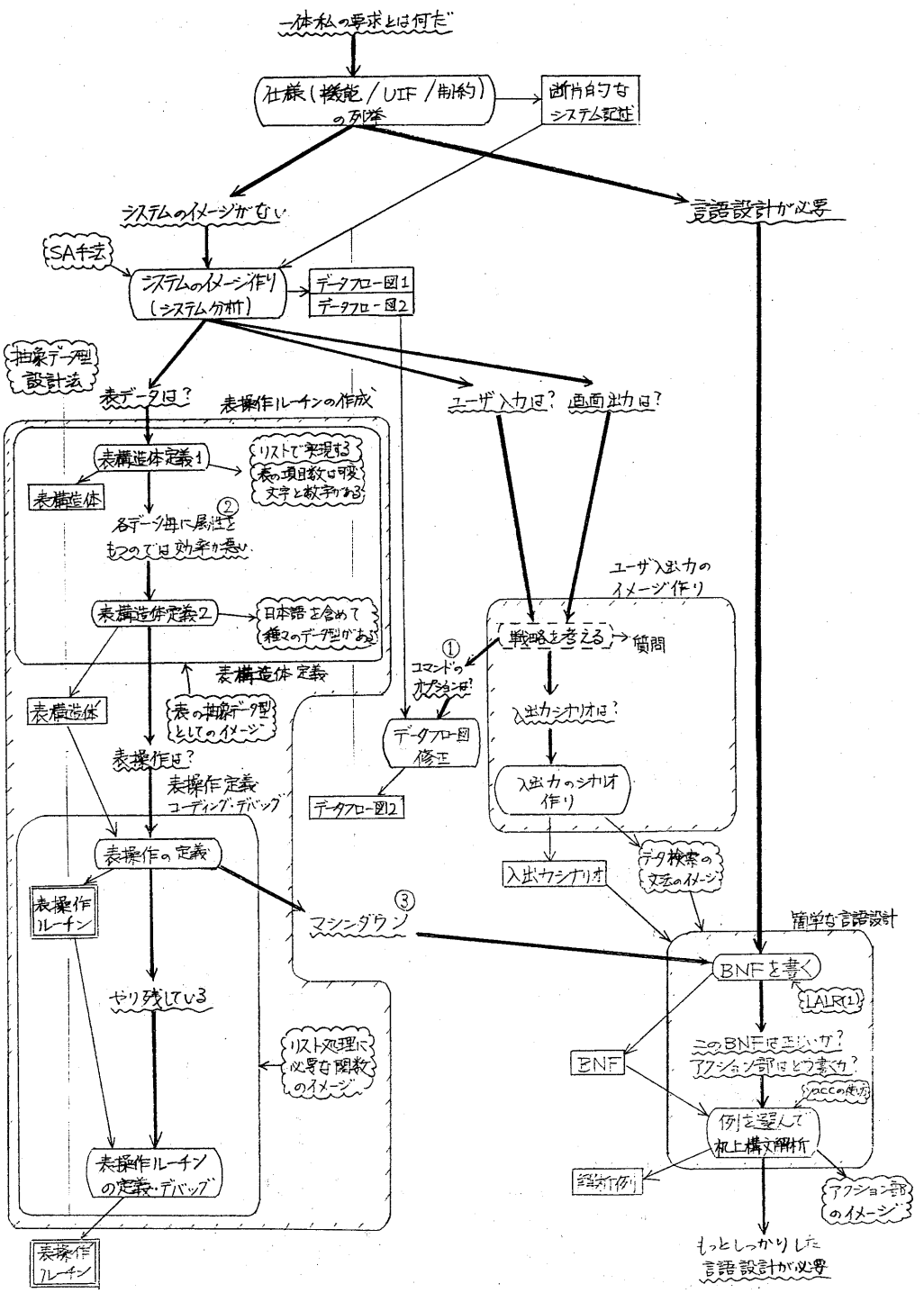



図６．設計プロセスの記述例（個人用簡易データベース）

4.1 設計プロセスと問題の発生原因

設計プロセスの良い悪いを議論する上で、問題発生の原因が重要である。三つの記述実験の結果、この原因には次の四つのタイプがあることがわかった。

(1) 外的要因

例えばマシンダウン・会議の時刻がきたなど。

(2) 思いつき

異なる問題を解いていてふと気がつく。

(3) 小問題への分解

問題を解き始める前により小さな問題に分解する。

(4) 手順の明確化

問題を解くための手順を予め生成する。

(1)は 除いて、(2) (3) (4) の順に、問題発生の洗練度が高くなる。(2)は 場当たり的、(3)は 発生する問題を事前に予想してはいるが解くための戦略がない、(4)は 問題の重要度・効率などを考え作業の順序を考えている。

さらに、個人差はあるが、多くの問題発生の原因は (2)に属しており、(2) の中に設計プロセスを重大に変えるものがあることもわかった。

設計プロセスを悪くする原因として、問題を解く手順を間違えるために起こる後戻りがある。これをできるだけ少なくするには、思いつきの問題を事前に見い出し、さらに問題を解くための後戻りのない手順を探す必要がある。

4.2 プロセスを良くする

プロセスを良くするには、外的な問題発生と単純なミスを削除することと、4.1 の考察から思いつきの問題発生を、効率的な手順に置き換えることが必要である。次の手順の内 (1) - (3) が前者、(4) - (6) が後者に相当する（図7）。図6の記述に (1) - (6) を適用した結果が図8である。

図8は、生産物が生成されていく流れを把握するための視野を与えてくれる [6]。

(1) 外的要因（図6③）を削除する。
(2) 見落としゆえに同じ作業が二度以上現れている箇所（図6①②）を探し削除する。
(3) 前の作業の入力知識に失敗した原因に関する情報を入れる（図8の①②）。
(4) 思いつきの問題を発生した作業から切り離す。
(5) 作業－生産物の連鎖に並べ変える。
(6) 依存関係から作業の順番を決める。

図7．プロセスを良くする手順

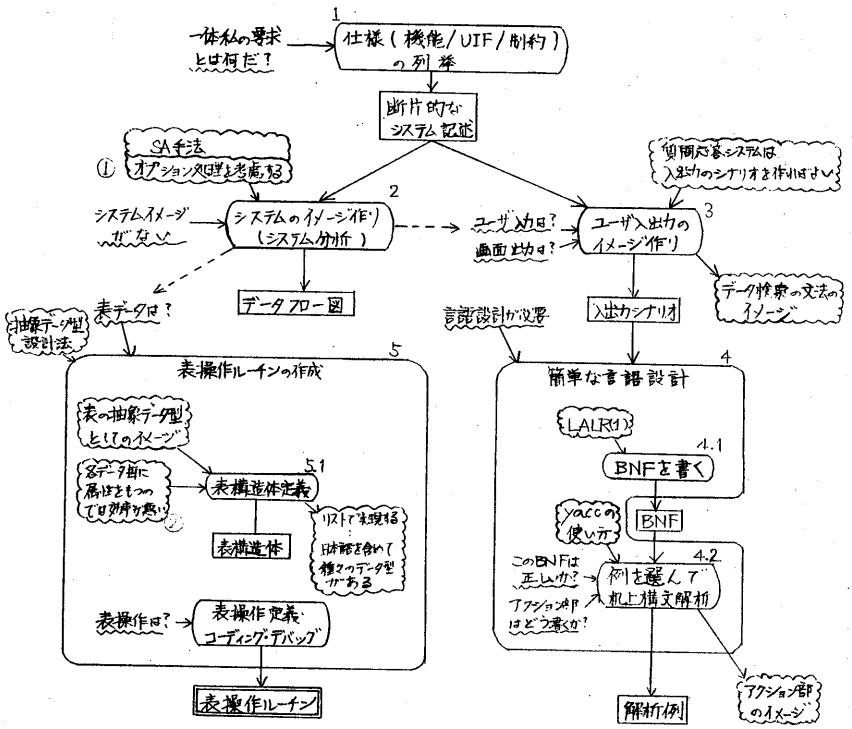



図8．図6を修正したプロセス

4.3 設計手法を抽出する

図9は、プロセスを良くする手順（図7）の後、実プロセスの記述から設計手法を抽出する手順である。図8から設計手法を導出した結果を表1に示す。

一般に、ＪＳＰやＳＡ／ＳＤなどの設計手法は、行うべき作業の手順と、個々の作業での入出力生産物（中間生成物）と、手順の分岐を与えるために必要な生産物の評価基準からなっている。ここでは、設計手法の厳密な定義には立ち入らずに、作業手順・入出力生産物・作業の終了条件の形式をもつものを設計手法と呼ぶことにする。

(1) 作業名を順に書き出す。
　そのとき、入力問題が作業の目的、入力知識が手段方法や失敗経験に相当するので、意味が通るように修飾する。
(2) 個々の作業に対して入出力生産物を書き出す。
(3) 出力知識から個々の作業の終了条件を抜き出す。

図9．設計手法を抽出する手順

表1．図8から抽出した設計手法の雛型

| 設計手順 | 入力生産物 | 出力生産物 | 終了条件 |
|---|---|---|---|
| 1 仕様の列挙<br>自分の要求を明確にするために機能・ＵＩＦ・制約について仕様を列挙する | | 断片的なシステム記述 | |
| 2 システム分析<br>システムイメージを作るために、ＳＡ手法を使ってシステム分析する。このときオプションの処理を考慮する。 | 断片的なシステム記述 | データフロー図 | 設計すべきデータが明らかになるまで |
| ＊データ定義<br>ＳＡ手法で明らかになったデータの設計を行う。<br>ユーザ入力・画面出力 → 手順3<br>表定義 → 手順5<br>3と 5は並列に行ってもよい。 | データフロー図<br>断片的なシステム記述 | 入出力シナリオ<br>表操作ルーチン | 手順3, 4, 5が終るまで |
| 3 ユーザ入出力のイメージ作り<br>ユーザ入力と画面出力を明確にするために入出力シナリオを書く。 | 断片的なシステム記述 | 入出力シナリオ | データ検索の文法のイメージが明らかになるまで |
| 4 簡単な言語設計 | 入出力シナリオ | ＢＮＦ・解析例 | |
| 4.1 ＢＮＦを書く<br>LALR(1)の知識に基づいて文法をＢＮＦで書く | 入出力シナリオ | ＢＮＦ | |
| 4.2 例を選んで机上構文解析<br>書いたＢＮＦが正しいことを確かめアクション部のイメージを作るために入出力シナリオから例を選んで机上構文解析する。 | 入出力シナリオ・ＢＮＦ | 解析例 | アクション部のイメージが明らかになるまで |
| 5 表操作ルーチンの作成<br>抽象データ型設計法に基づき、表データを設計し操作を定義する。 | | 表操作ルーチン | |
| 5.1 表構造体定義<br>表の抽象データ型としてのイメージを基に、表構造体を作る。<br>このとき属性のもち方に注意する。 | | 表構造体 | |
| 5.2 表操作定義・コーディング・デバッグ<br>表操作を定義するために、表操作ルーチンをコーディング・デバッグする。 | 表構造体 | 表操作ルーチン | アクション部のイメージが明らかになるまで |

4.4 抽出した設計手法の評価

抽出した設計手法の雛型（表１）は、次の点が不明確である。

(1) 記述の形式性

設計者が自由な手法を用いる場合、設計の中で記録された生産物の文法と意味は曖昧になる。例えば、「ＢＮＦ」は文法や意味付けが明確であるが、「断片的なシステム記述」は自然言語の羅列である。同じ理由で、作業の終了条件も曖昧である。

(2) 適応範囲

基となる実プロセスは、設計すべきソフトウェアの性質と、設計に当たっての制約（仕様の十分性等）の影響を大きく反映している。従って、抽出した設計手法の適応範囲は限られており、注意が必要である。この例では、「仕様が曖昧に頭の中にあるような個人向き簡易データベースの設計」という条件をつければ適用できる部分が多い。

(3) 価値

基となる実プロセスに変更は加えてあるが、なお出来上がった設計自体が悪いものであれば、設計プロセスもまた悪い。従って、最低設計自体の品質が保証されている必要がある。

上記の問題を解決して、意味のある設計手法を導くようにするためには、次のアプローチが必要であると考えている。

(1) 生産物に文法と意味が定義された図法を使う

さらに設計者がその図法をよく理解した上で使うことが必要である。

(2) 適応範囲を明記し、多くの設計手法を集める

適応範囲の近い多くの例を比較することで、高いレベルの抽象化を生む可能性がある。

(3) よい品質を生んだ実設計プロセスだけを選ぶ

## 5．まとめ

ＰＰＫモデルで記述された設計プロセスは、理解しやすく分析しやすい性質を持っている。これは、このモデルが、簡潔であるにも関わらず、設計プロセスの意味を表現できるからである。本報告では、設計手法の抽出について論じたが、これは、モデルの分析能力の高さを示している。

今後は、より多くの実設計のプロセス記述を通じて、プロセスの評価技術・再利用技術・支援技術を高めていくと同時に、手法自体の改良も進めていくつもりである。

## 参考文献


[1] 1st-4th 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] J. Conklin and M. Begeman, "gIBIS: A Tool for All Reasons," MCC Technical Report, No. STP-252-88

[5] J. Conklin and M. Begeman, "gIBIS: A Hypertext Tool for Exploratory Policy Discussion," MCC Technical Report, No. STP-082-88

[6] 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

[7] 川喜田二郎，発想法，中公新書