ソフトウェア設計プロセス構成法の一提案

望月純夫† 片山卓也††

ソフトウェア設計技術の伝承は，ソフトウェア開発部門にとって重要な課題である．この目的を達成するためには，設計技術を顕在的に記述することが必要である．我々は，過去に設計を経験したシステムのオブジェクトおよび手順をプロセスモデルHFSPを基にして分析し，部門の標準的な設計プロセスを明らかにした．その結果，一つのシステムの設計プロセスを表現する場合，その視点を変えることによりオブジェクト中心型プロセスとフェイズ中心型プロセスの二つの表現方法があることを発見した．熟練技術者は，自分の設計経験から普遍的な設計プロセスであるオブジェクト中心型プロセスを抽出して蓄積し，それを設計対象システムに合わせたフェイズ中心型プロセスに変換して設計に適用していると考えられる．これら二つのプロセス表現を比較した結果オブジェクトの遷移が重要な鍵であることを発見した．本論文ではこれを基にして，前述の二つのプロセス表現を論じた．実際の設計で作成されたオブジェクトを分析し，それを基にして具体的なシステムを設計する時のフェイズの構成法を示した．さらに，そのフェイズの中でオブジェクト中心型プロセスをどのように実行するかを示し，これによってフェイズ中心型プロセスの構成法を示した．この研究により，これまで曖昧であったフェイズの作成方法を明らかにした．

A Proposal for Composing Software Design Process

SUMIO MOCHIZUKI † and TAKUYA KATAYAMA ††

It is an important theme for software company to hand down their software engineering. For this purpose it is necessary to clarify and describe design techniques. We wanted to describe formally design process of our experienced engineers, and analyzed the objects and the procedures in real design on the basis of process model HFSP. In our reseach we discovered two types of describing methods of design process.: Object-centered process and phase-centered process. The experienced engineer converts the generic object-centered process into phase-centered process, when he designs a real system. In this paper we described those two processes on the basis of object transition. We analyzed the real objects which are produced in the real design, and showed the method of composing phases. We also described the way how the object-centered process is executed in the phases, and the way of composing phase-centered process.

1. はじめに

ソフトウェア開発部門にとってソフトウェア技術，特にソフトウェア設計技術の伝承はきわめて重要な課題である．最近，このようなソフトウェア設計技術を顕在的に記述するための技術としてソフトウェア・プロセスが研究されており[1)~3),7)]，その改善努力も続けられている[4)]．

筆者の所属していた企業では，これまで数多くの実時間処理システムの設計を手掛けており，その標準的な設計技術を記述して技術の伝承に利用したいと考え，プロセス・モデルHFSP (Hierarchical and Functional Software Process) [5)]に基づく設計作業の分析，記述に取り組んできた[6),8)~10)]．設計技術を分析する対象システムとしては，我々が豊富な設計経験を持つ実用システムの一つ『人工衛星のチェックアウトを目的としたデータ収集・解析システム (Data Acquisition System，以下DASと略称する)』を選んだ．

この研究の中で，基本設計段階における熟練技術者の設計プロセスを分析した結果，一つのシステムの設計プロセスを表現する場合にその視点を変えることによってオブジェクト中心型プロセスとフェイズ中心型プロセスの二つの表現方法があることを発見した．熟練技術者は自己の豊富な設計経験から普遍的な設計プロセスを抽出して蓄積し，それを設計対象システムに

† 淑徳短期大学
Shukutoku Junior College
†† 東京工業大学情報理工学研究科計算工学専攻
Department of Computer Science, Graduate School of Information Science and Engineering, Tokyo Institute of Technology

合わせたプロセスに変換して実際の設計に適用していると考えられる．この二つのプロセスについては文献9)および10)に示した．しかし，それらの変換については述べていない．

この普遍的な設計プロセスは，一つの応用分野に属するシステム設計を共通の一つのプロセスで表現することができるものであり，オブジェクト中心型プロセスにより記述することができる．しかし，一般にはこのプロセスはそのままでは実行することができず，設計対象システムに適合するフェイズ中心型プロセスに変換する必要がある．

これらの二つのプロセスの比較をしてみると，オブジェクトの遷移が重要な鍵であることを発見した．本論文では，これらのことからフェイズ中心型プロセスの作り方を中心として次の内容を論じている．

（1）典型的な二つのオブジェクトの遷移があることを述べた．

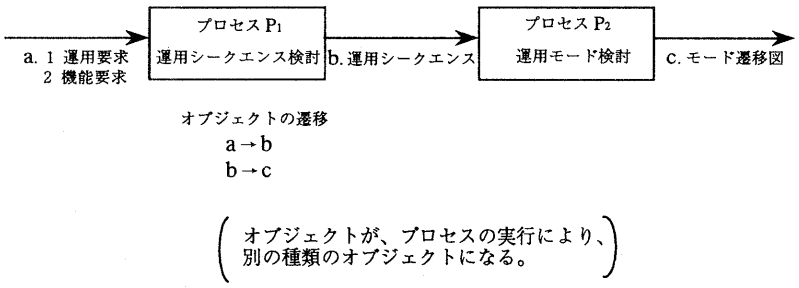
（2）プロセスの繰り返しと先読みという概念を導入してフェイズ中心型プロセスの構成法を明らかにした．

（3）フェイズ中心型プロセスが実際の設計プロセスとして適切であることを確認した．

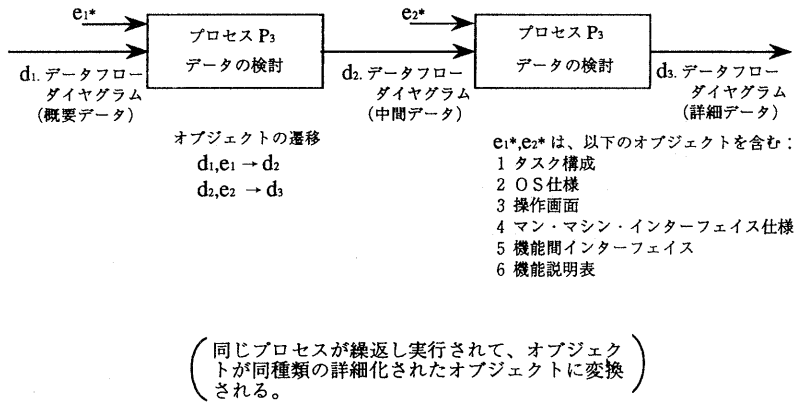
2. 階層的関数型プロセス・モデル HFSP とオブジェクトの遷移

2.1 プロセス・モデル HFSP[5]

ソフトウェア・プロセスには，様々な側面がある．その中で，HFSP の最も基本的なものは機能的側面であり，仕事の単位であるプロセスによって作成されるオブジェクト（要求仕様書，設計情報，関連資料，技術的データ等をオブジェクトと総称する）の関係に着目することである．ソフトウェア設計においてソフトウェア・プロセスをその要素プロセスの集合と考え，プロセスをその入出力オブジェクト間の関数として捉える．プロセスは，オブジェクトを加工して次のプロセスに渡し，さらに次のプロセスがそれを加工する．



(a) I 型遷移



(b) II 型遷移

図1 プロセス実行による二つの典型的なオブジェクト遷移
Fig. 1 Two typical transition types of objects.

得られたオブジェクトは，最終的に設計書としてまとめられる．

プロセス・モデル HFSP では，このように設計手順であるプロセスを数学的な関数により詳細かつ厳密に表現する．プロセスの機能が複雑である場合は，そのプロセスを更に細かいサブ・プロセスに分解して第2の階層として，各サブ・プロセス間の入出力オブジェクトを明確に定義する．この作業を各々のプロセスが十分に単純化されるまで続ける．この関数関係は階層的な分割を通して定義されるがこれを機能的側面と呼んでいる．機能的側面は，オブジェクト中心型プロセスの記述に適している．

HFSP では，この機能的側面のほかに動的側面および実働的側面があり，その三つの独立な側面によってソフトウェア・プロセスを記述しようとしている．動的側面では，それらの関数の起動される順序やプロセスの作成などのような動的な事柄を規定する．これは機能的側面がオブジェクト間の静的な関数関係を規定しているのと対照的である．この動的側面を用いてフェイズ中心型プロセスを記述することができる．一方，

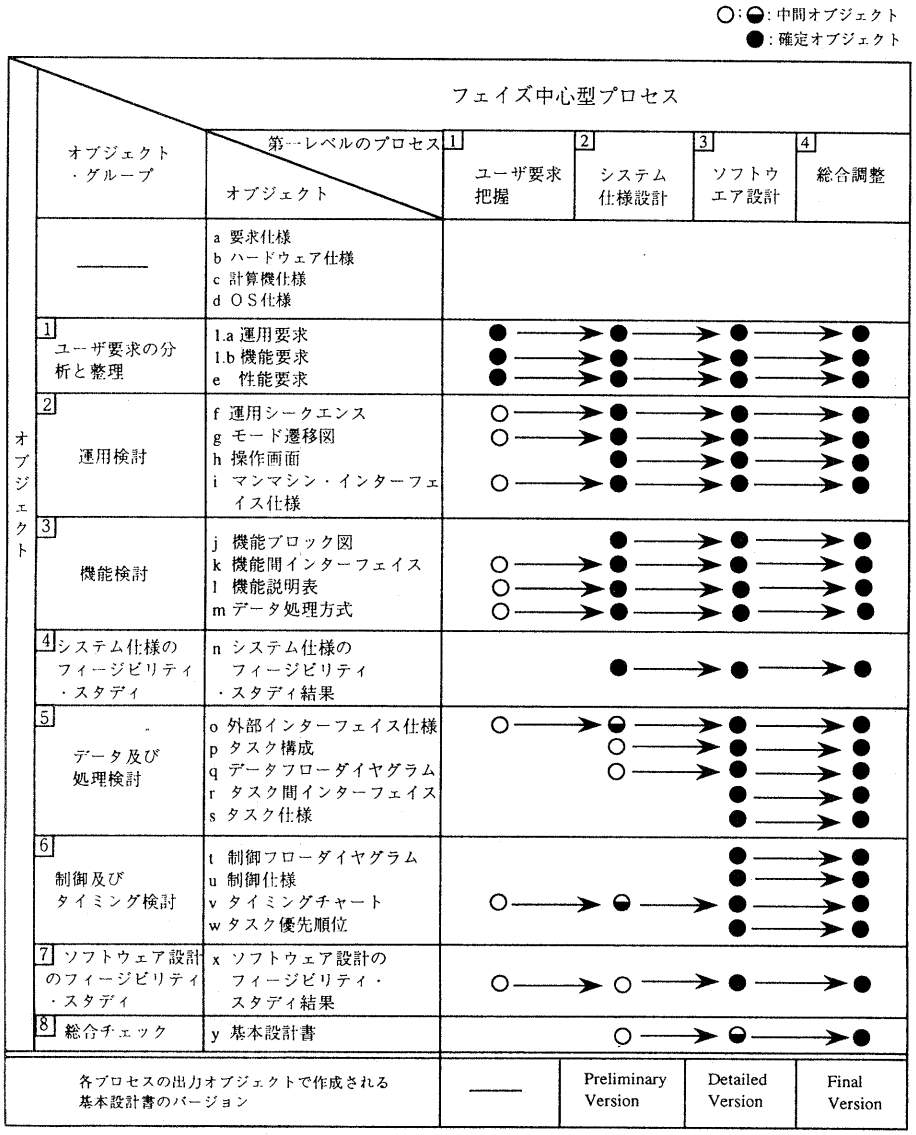


図2 実用システム DAS 設計のためのフェイズ中心型プロセス（第1レベル）の各プロセスによるオブジェクト遷移図

Fig. 2 Transition diagram of the objects by executing first level phase-centered process for designing the practical system DAS.

実働的側面では，機能的および動的側面で記述されたことが実際の組織やプロジェクトで実現されるための機構が規定され，資源管理，スケジュール管理などが含まれる．

2.2 オブジェクトの加工とその遷移

設計の過程では様々な種類のオブジェクトが作成され加工されていく．設計の途中段階で作られる大部分のオブジェクトは次のプロセスによってさらに加工される一時的なオブジェクトであり，これを中間オブジェクトと呼ぶ．一部のオブジェクトはそのまま設計の最後まで残り，設計書としてまとめられる．これを確定オブジェクトと呼ぶ．設計作業の目的は，最終的に相互に矛盾の無い確定オブジェクトのセットを求めてこれらをもとにして設計書をまとめることである．

オブジェクトは，プロセスの実行を通して加工されるが，その履歴をオブジェクトの遷移と呼ぶ．筆者らが設計を分析したDASシステムのオブジェクトを調べてみると設計過程におけるオブジェクトの遷移には**図1**に示すようなI型遷移とII型遷移の二つの型があることが判明した．図中ではプロセスをブロックとして示し，その実行順序を矢印で示している．各プロセスの入力オブジェクトおよび出力オブジェクトをそのブロックの前後に示す．

①I型遷移

入力オブジェクトがプロセスによって処理されて，異なる種類の出力オブジェクトに変換される場合である．図中では，オブジェクトa(運用要求および機能要求)がプロセスP_1(運用シークエンス検討)およびプロセスP_2(運用モード検討)の実行によって処理されて，次の順序で各々異なる種類のオブジェクトに変換される．

a（運用要求）
（機能要求）} → b（運用シークエンス）
→ c（モード遷移図）

②II型遷移

入力オブジェクトまたはその一部が出力オブジェクトと同じ種類のオブジェクトである場合である．規模の大きなシステムの設計では，膨大なオブジェクトが作成されるため，最終的に一貫したオブジェクトをまとめるまでには何回も同じオブジェクトを求めて粗いオブジェクトから次第に詳細なオブジェクトに変換する．図中ではオブジェクトd(データフローダイヤグラム)がプロセスP_3(データの検討)の繰り返しによって，次のように次第に詳細化されている．

d_1（概要データ）→ d_2（中間データ）
→ d_3（詳細データ）

ここに述べたオブジェクトのI型遷移とII型遷移の例として実用システムDAS設計の実行に伴ったオブジェクトの遷移を**図2**に示す．この図では左から右の順にフェイズ中心型プロセスが実行されると共に遷移するオブジェクトの状態を3段階の粗い表示としており，●印は，そのオブジェクトがほぼ確定されていることを表し，○印は未確定オブジェクトを示し，◒は○と●の中間的なオブジェクトである．矢印は各フェイズ中心型プロセスの各プロセス間のオブジェクトのII型遷移を示している．

3. 設計プロセスの記述を目的とした二つのモデル

これまでの研究で，プロセス・モデルHFSPのプロセスの定義と構造に基づいて現場における実用システムの基本設計段階の設計プロセスを記述し，評価してきた[6),8),9)]．我々が何回も設計を経験した実用システムの一つDASを選び，その設計手順を実際の設計作業の順序に忠実に記述した．これは本論文で述べるフェイズ中心型プロセスにあたる．

これを別の熟練技術者のグループに評価させた結果，このソフトウェア・プロセスは現場における実用システムの設計作業を忠実に表現してはいるが，設計過程の中で同じ種類のプロセスおよびオブジェクトが何度も現れるため，他の技術者には理解しにくいことが判明した．

次に，一つのシステムの全オブジェクトを収集し，**表1**のようにその内容の類似性により八つのグループに分類した．さらに各々のオブジェクト・グループに対して，それを出力オブジェクトとするプロセスを定義した．これは本論文のオブジェクト中心型プロセスにあたるものである．

ここに得たフェイズ中心型プロセスとオブジェクト中心型プロセスは，一つのシステムの設計プロセスを別の角度から記述するものである．以下にこれら二つのプロセスを定義する．

3.1 オブジェクト中心型プロセス

一つの実用システムの基本設計段階では多くのオブジェクトが作成されるが，このオブジェクトの中には，図1で示したII型遷移によって生ずる同じ種類のオブジェクトが多数含まれている．オブジェクト中心型モデルでは，同じ種類のオブジェクトを一つのオブジェクトとして取り扱うこととする．例えば，図1(b)のd_1, d_2, d_3は，一つのオブジェクト「データフローダイヤグラム」として考える．この操作の結果，残ったオブジェクトは，I型遷移によって生ずるオブジェクト，

表1 基本設計段階で作成されるオブジェクトとそのグループ分類
Table 1 Classification of objects produced in fundamental design.

オブジェクトグループ	オブジェクト		
	オブジェクト	中間オブジェクト*	オブジェクト数
—	a. 要求仕様 b. ハードウェア仕様 c. 計算機仕様 d. OS 仕様	—	4
1.ユーザ要求の分析と整理	e. 性能要求	1.a 運用要求 1.b 機能要求	3
2.運用検討	f. 運用シーケンス g. モード遷移図 h. 操作画面 i. マンマシン・インタフェース仕様	2.a 操作イメージ 2.b 操作シーケンス 2.c 画面遷移	7
3.機能検討	j. 機能ブロック図 k. 機能間インタフェース仕様 l. 機能説明表 m. データ処理方式	3.a 機能概要 3.b 機能間関係説明	6
4.システム仕様のフィージビリティ・スタディ	n. システム仕様のフィージビリティ・スタディ結果	—	1
5.データおよび処理検討	o. 外部インタフェース仕様 p. タスク構成 q. データフローダイヤグラム r. タスク間インタフェース仕様 s. タスク仕様	5.a タスク間インタフェース方法 5.b 共有メモリ仕様 5.c データファイル仕様 5.d 外部入出力データ仕様 5.e タスク仕様	10
6.制度およびタイミング検討	t. 制御フローダイヤグラム u. 制御仕様 v. タイミングチャート w. タスク優先順位	6.a 制御フロー 6.b 制御条件	6
7.ソフトウェア設計のフィージビリティ・スタディ	x. ソフトウェア設計のフィージビリティ・スタディ結果	—	1
8.総合チェック	y. 基本設計書	—	1

*中間オブジェクトは，他のオブジェクトを作成するために一次的に作られるオブジェクトである．

すなわち異なる種類のオブジェクトのみとなる．このようにして得られたオブジェクトのうち，密接な関係にあるオブジェクトを各々一つのグループとした．その結果最終的に表1に示す次の八つのオブジェクト・グループを得た．

①ユーザ要求の分析と整理
②運用検討
③機能検討
④システム仕様のフィージビリティ・スタディ
⑤データおよび処理検討
⑥制御およびタイミング検討
⑦ソフトウェア設計のフィージビリティ・スタディ
⑧総合チェック

次に**図3**に示すように各オブジェクト・グループを出力するプロセスを定義する．すなわち，任意の一つのオブジェクト・グループに対して，それを出力する一つのプロセスを定義し，さらにそのオブジェクト・グループ内の各オブジェクトを作成するために必要なオブジェクトをそのプロセスの入力オブジェクトとした．この方法で前述のオブジェクト・グループに対してプロセスを定義すると**図4**に示すソフトウェア・プロセスを得る．これを第1レベルのオブジェクト中心型プロセスと呼ぶ．このプロセスの内容はマクロ的すぎるため，各プロセスをさらに詳細化して第2レベル

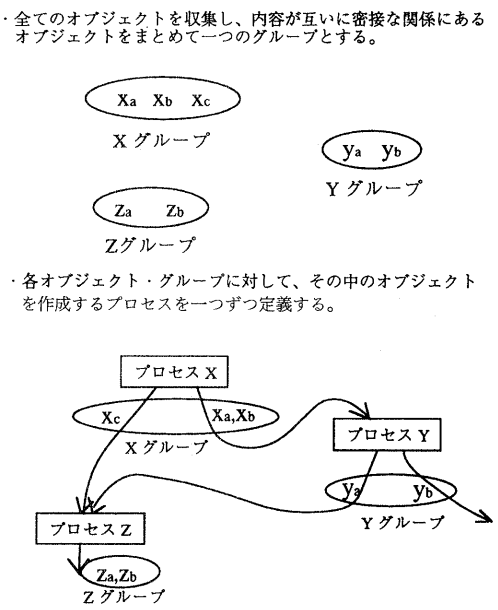


図3 オブジェクト中心型プロセス
Fig. 3 Object-centered process.

のプロセスを得る．この結果42個のオブジェクトと17個のプロセスを含む（詳細については文献9)参照）第1レベルと第2レベルの2階層のオブジェクト中心型プロセスが得られた．

このオブジェクト中心型プロセスでは，密接な関係のあるオブジェクトをオブジェクト・グループとしてまとめており，各オブジェクト・グループごとにそれを作成する一つのプロセスを設定している．さらに，オブジェクトの遷移はI型遷移のみで構成しているため，このプロセスはオブジェクトの間の静的な依存関係を簡潔にわかりやすく表現することができる．また同じ応用分野に属するシステムの設計手順をこのオブジェクト中心型プロセスによって記述すると一つの共通なプロセスとして表現できるため，このプロセスはその応用分野の普遍的モデルであるといえる．

3.2 フェイズ中心型プロセス

オブジェクト間の静的な依存関係を重視したオブジェクト中心型プロセスに対して，フェイズ中心型プロセスは，具体的なシステムを設計する場合の現実のプロセスをその実行順序に忠実に記述しようとするものである．

非常に単純なシステムを設計する場合，例えば計測点が1点でデータの種類が1種類である計測システムの場合にはオブジェクトの量も少なく，そのオブジェクト中心型プロセスをそのまま実行すれば順調に設計を進めることができる．しかし，実用システムのような規模の大きいシステムを設計する場合には，論理的に深いオブジェクトを作らなければならないので，単純にオブジェクト中心型プロセスを一通り実行するだけではオブジェクトの中に完成度の異なるものが生ずる．これらを含むオブジェクトは互いに依存関係にあり，このままではそれ以後のプロセスが実行できないため，既に実行したプロセスに戻って同じプロセスを繰り返し実行し，オブジェクトの完成度を高める．このようなプロセスの繰り返しを行いつつ，少しずつ先へ進める．この過程でオブジェクトのII型遷移が発生する．

プロジェクト管理の立場からみると，このような設計作業を秩序立てて進めるためには作業をいくつかのフェイズに区切り，フェイズごとにオブジェクト間の整合性をとる必要がある．このフェイズの定義は次のとおりである．

（1）各フェイズの作業の目的は，そのフェイズに定められたオブジェクトを確定することである．
（2）フェイズの実行に伴ってオブジェクトが次々に確定され，原則としてフェイズ間の手戻り作業はない．
（3）依存関係の強いいくつかのオブジェクトは，一つのフェイズの中でまとめて確定する．
（4）システムの規模が大きくなるとプロセスの繰り返しおよび先読み実行が必要となる．
（5）フェイズは設計作業の区切りであり，その最後には，工程，コスト等のレビューを含むプロジェクト管理とのインタフェースを持つ．

すなわち，各フェイズに定められたオブジェクトを確定することがそのフェイズの目的であり，この目的とするオブジェクトを確定するためにはフェイズの中でオブジェクト中心型プロセスを何回か繰り返す必要があり，またオブジェクトの裏付けを行うために先読みを行う．これらの作業によってフェイズ間の手戻りの発生を避けている．この繰り返しと先読みは次のとおりである．

（1） 大きな手戻りの防止を目的とした作業の繰り返し

矛盾の無い大量のオブジェクトを効率よく作成するために，フェイズ内部のプロセスを繰り返し実行して，得られたオブジェクトの矛盾を調整する．

（2） 入力オブジェクトが少しずつあたえられる場合の設計作業の繰り返し

一般に，実用システム設計では必要な入力オブジェクトが一挙に揃わず少しずつ与えられる場合が多い．この時，入力オブジェクトが揃うまで設計作業

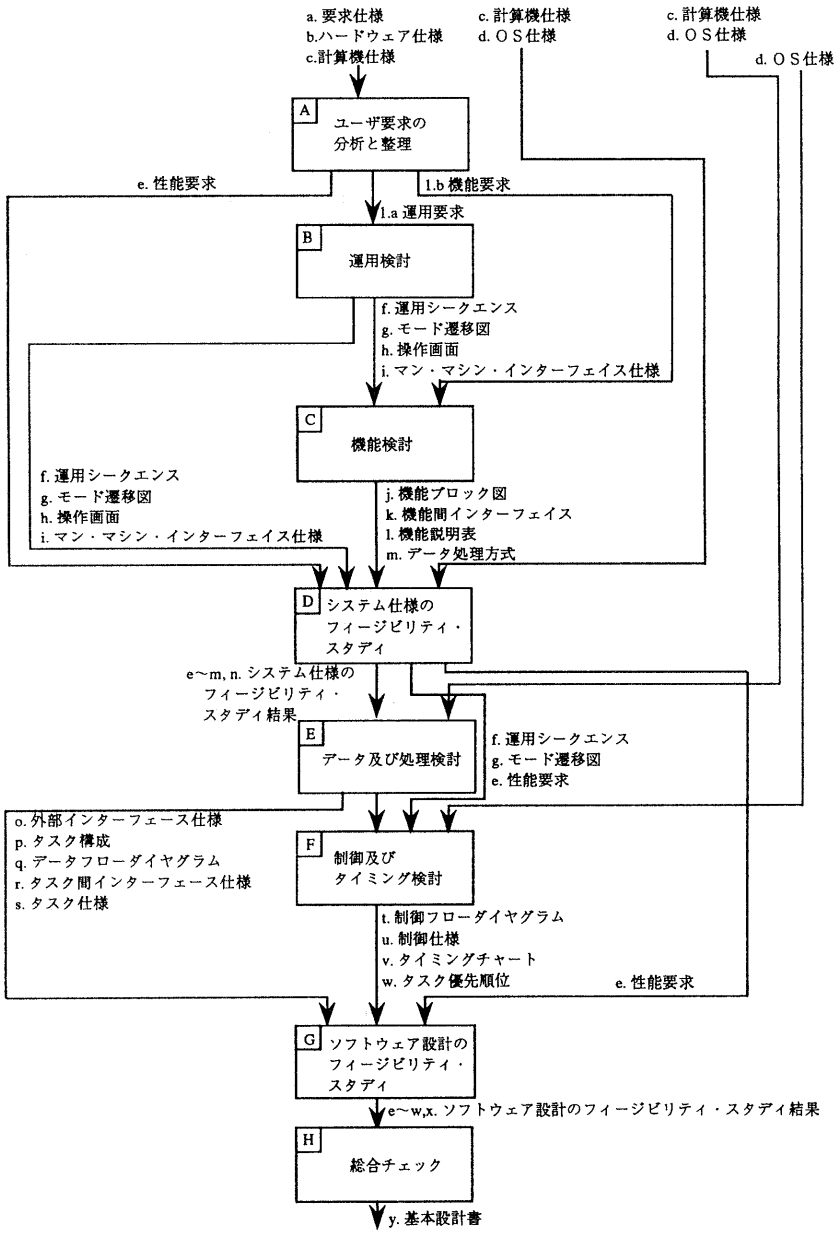


図4 DASシステムの基本設計のオブジェクト中心型プロセス
(第1レベル)

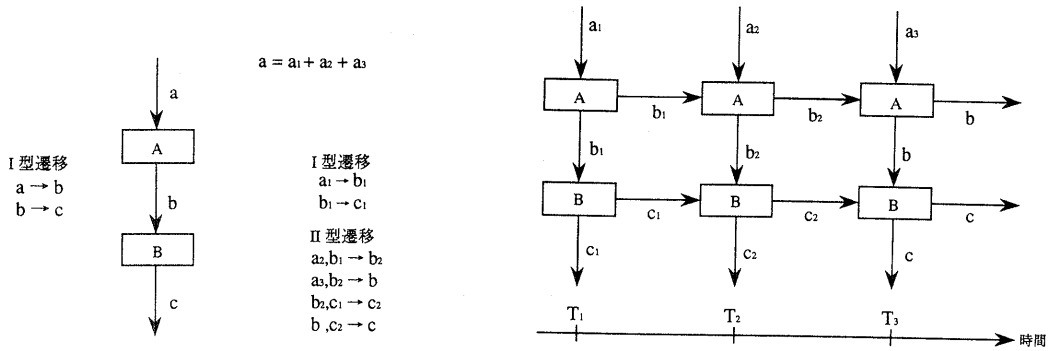
Fig. 4 First level object-centered process for fundamental design of DAS system.

を中止すると工程遅れとなるため，不足するオブジェクトを仮定して設計作業を進め，入力オブジェクトの追加と共に設計作業を繰り返す．

(3) プロセスの先読み

一つのフェイズの中で得た確定オブジェクトの裏付けをするため，本来その後のフェイズで実行されるべきプロセスをそのフェイズの中であらかじめ実行し，これらのオブジェクト間の矛盾を除去する．これをプロセスの先読みと呼ぶ．この先読みの結果，隣合うフェイズ間で，同じプロセスを繰り返すことになり，フェイズ間のオブジェクトのII型遷移が生ずる．

フェイズ中心型プロセスでは，フェイズを設定し，各フェイズの中では上記(1)(2)の理由によりオブジ



（a） オブジェクト中心型プロセス　　（b） 実際の設計プロセス

図5 オブジェクト中心型プロセスと実際の設計プロセスとの比較
Fig. 5 Comparison between object-centered process and real design process.

ェクト中心型プロセスの一部を繰り返し実行する．その結果オブジェクトは前述のII型遷移によって遷移し，プロセスの繰り返しと共にその内容が少しずつ詳細化される．

図5は，(2)の入力オブジェクトが少しずつ与えられる場合の例として，入力オブジェクト a が時間の経過と共に少しずつあたえられる時の実際のプロセスの実行とオブジェクトの遷移を示している．図5(a)は，オブジェクト中心型モデルを表し，図5(b)は，実際にそれを実行する時の設計プロセスを示している．プロセスAの入力オブジェクト $a(=a_1+a_2+a_3)$ は，時間の経過 T_1, T_2, T_3 と共に a_1, a_2, a_3 と部分的に少しずつ与えられる．入力オブジェクトが与えられるにともなって，プロセスA，Bが繰り返され，オブジェクト a のすべての部分が揃った時点 T_3 で初めてオブジェクト b，c を確定することができる．

フェイズ中心型プロセスは，実際のプロセス実行の順序に基づいて表現しており，実作業にそのまま利用できるという利点がある．しかし，実際の設計プロセスにおける前述の繰り返しや先読みはシステムの規模によっても異なるため，一つのフェイズ中心型プロセスの利用可能範囲は同じ応用分野に属するほぼ同じ規模のシステムの設計に制限されることになる．

4. フェイズの構成法

システムを実際に設計するに当たりオブジェクト中心型プロセスをそのシステムに適合するフェイズ中心型プロセスに変換する必要がある．その変換の手順は次のとおりである．

①各フェイズで確定すべきオブジェクトの決定

非常に深い関係のあるオブジェクトは一つの同じフェイズで確定する必要がある．オブジェクト・グループの間の相互の関係を分析し，関係の深いオブジェクト・グループは一つにまとめて，一つ上位の階層のグループを作成する．

②基本フェイズの構成

この上位のグループを作成するプロセスとして，各グループに一つづつ基本フェイズを割り当てる．各基本フェイズに対応するオブジェクトを作成するためのプロセスとしてオブジェクト中心型プロセスの一部を割り当てる．

③基本フェイズを基にしたフェイズの構成

設計対象システムの設計に基本フェイズを適用したときの難易度を評価する．適切な難易度となるように基本フェイズを要素としてフェイズを構成する．

このようにして最終的に得られたものがそのシステムの設計のためのフェイズ，すなわちフェイズ中心型プロセスの第1レベルのプロセスである．

4.1 基本フェイズの作成

表1のオブジェクト・グループを観察すると複数のオブジェクト・グループの間で互いの関係が深いものと，そうでないものとがある．設計作業の手戻りを少なくさせるために関係の深いオブジェクト・グループは一つのフェイズで確定する．そのほかのオブジェクト・グループは各々別のフェイズで確定する．この様にして各フェイズで確定すべきオブジェクトを編成して，その各々に対して一つずつフェイズを割り当てる．求められたフェイズは，それ以上分解することのできないフェイズの最小単位であり，これを基本フェイズと呼ぶ．表1ののオブジェクト・グループの集合を分解して，関係の深さでまとめて，その各々に対して基本フェイズを割り当てると次のようになる．

(関係の深いオブジェクト・グループをまとめる.) → (基本フェイズを割り当てる.)

- ユーザ要求の分析と整理 →ユーザ要求把握
- 運用検討
- 機能検討
- システム仕様のフィージビリティ・スタディ →システム仕様設計
- データおよび処理検討
- 制御およびタイミング検討
- ソフトウェア設計のフィージビリティ・スタディ →ソフトウェア設計
- 総合チェック →総合調整

この中で,『運用検討』,『機能検討』および『システム仕様のフィージビリティ・スタディ』の各オブジェクト・グループのオブジェクトは,システム仕様を定めるものとして互いに密接な関係にあり,一つのオブジェクトの変更は他のオブジェクトに影響を与えて大きな手戻り作業を発生させる可能性を生ずるため,同一のフェイズで確定されるべきものである.

同様に『データおよび処理検討』,『制御およびタイミング検討』および『ソフトウェア設計のフィージビリティ・スタディ』の各オブジェクト・グループのオブジェクトは,全体としてソフトウェア設計結果を表しており相互に密接な関係にあるので,一つのフェイズで確定されるべきものである.

このほかの『ユーザ要求の分析と整理』のグループと『総合チェック』のグループのオブジェクトは,各々定められた順序で単独で確定することができる.

これまでに述べてきたオブジェクト・グループは,図4のオブジェクト中心型プロセスをシークエンシャルに実行することにより作成されるが,プロセスがlattice状であっても同様に基本フェイズを構成することができる.また,複数のプロセスが平行動作をする場合には,それらのプロセスにより作成されるオブジェクトを同じ基本フェイズの中で確定するように設定する.

4.2 フェイズの構成

フェイズは,設計作業を最も効率良く進めることの

表2 基本フェイズと規模の異なるシステムのフェイズ
Table 2 Fundamental phases and phases for designing different-scaled systems.

オブジェクト・グループ	基本フェイズ	フェイズ		
		大規模システムのフェイズ	実用システム(DAS)のフェイズ	小規模なシステムのフェイズ
ユーザ要求の分析と整理	ユーザ要求把握	ユーザ要求把握(1) ユーザ要求把握(2) : :	ユーザ要求把握	要求分析
運用検討	システム仕様設計	システム仕様設計(1) システム仕様設計(2) : :	システム仕様設計	
機能検討				
システム仕様のフィージビリティ・スタディ				
データおよび処理検討	ソフトウェア設計	ソフトウェア設計(1) ソフトウェア設計(2) : :	ソフトウェア設計	ソフトウェア設計
制御およびタイミング検討				
ソフトウェア設計のフィージビリティ・スタディ				
総合チェック	総合調整	総合調整	総合調整	—

できる大きさに定める必要がある．非常に簡単なシステムを設計する場合には，基本フェイズをいくつかまとめて適切な作業量のフェイズを構成する．また，非常に複雑なシステムでは，基本フェイズをさらに細分化することはできないので基本フェイズを繰り返すことになる．実用システムDASの設計では，基本フェイズをそのままフェイズとして利用している．

具体的なシステム設計のフェイズを構成しようとするとき，この基本フェイズを基準として検討する．まず各基本フェイズに対応するオブジェクト中心型プロセスを割り当てる．すなわち，**表2**の四つの基本フェイズ(『ユーザ要求把握』，『システム仕様設計』，『ソフトウェア設計』，『総合調整』) に対応するオブジェクト・グループを作成するためのオブジェクト中心型プロセスを各基本フェイズのプロセスとする．その結果を**図6**に示す．次にこれらのフェイズが実際の設計作業として適切な作業量であるか否かを難易度により判定する．この難易度の定量的な計測は困難であるが，実際の設計では次のようにフェイズの適切さを難易度によって判定していると思われる．

(1) フェイズの重み

プロセス a の難易度を表す重みを $w(a)$ とする．この $w(a)$ の値を正確に定めることは困難であるが，そのシステムの規模と複雑さから経験的に定めることができる．

一つのフェイズの重みはそのフェイズを構成するプロセスの重みを総合したものであるから，それらのプロセスの重みを合計したものとする．例えば，フェイズ p の重み $W(p)$ は，フェイズ p を構成するプロセス a, b, c の重み $w(a), w(b), w(c)$ の合計で与えられる．

$$W(p)=w(a)+w(b)+w(c)$$

(2) 基本フェイズの難易度の判定とフェイズの再構築

実行可能な一つのフェイズの重みには上限値 (W_{max}) と下限値 (W_{min}) があり，フェイズの重みがこの範囲を越えた場合，そのまま設計に利用するには適切でないと判定されるため再構築の必要がある．ここで基本フェイズ q の難易度をその重み $Wb(q)$ によって判定し，その結果によりそのシステムの設計に適切なフェイズの構築を次のように行う．

① $W_{min} < Wb(q) < W_{max}$ の場合

この基本フェイズ q の作業量は妥当であり，そのまま設計のためのフェイズとして利用することができる．

② $W_{max} < Wb(q)$ の場合

この基本フェイズの作業は重みが大きすぎるが，

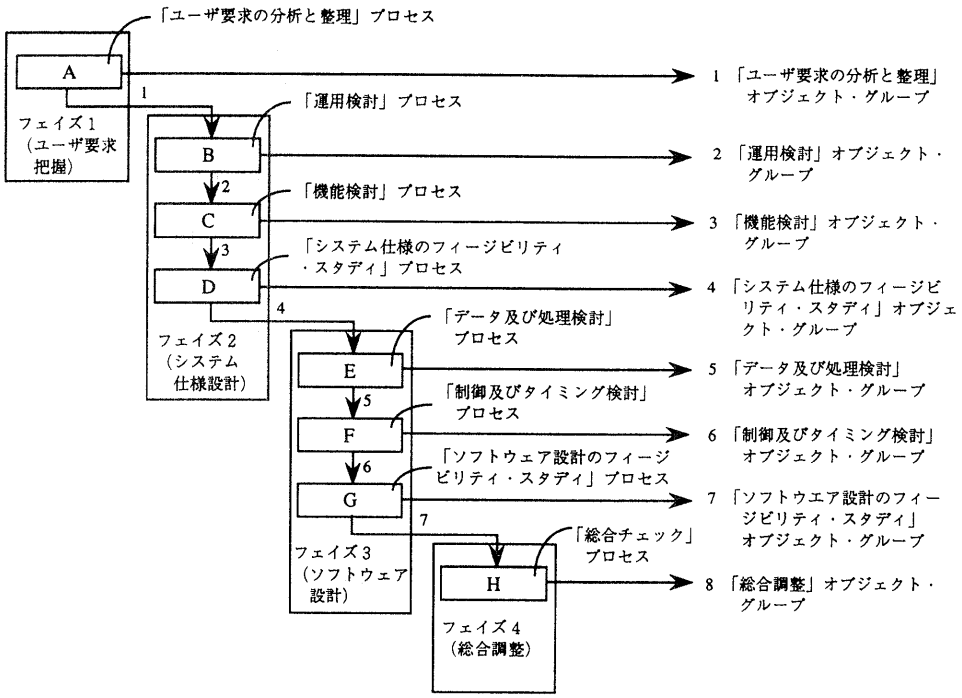


図6 基本フェイズとその目的とするオブジェクト，およびそれを確定するためのプロセス
Fig. 6 Fundamental phases, the objects defined in them, and the processes which produce the objects.

基本フェイズはそれ以上分割できないため，表2の大規模システムのフェイズとして示すように，基本フェイズ q をフェイズ $q(1)$，フェイズ $q(2)$，…のように繰り返す．このフェイズの繰り返し実行と共に次第にオブジェクトが詳細化され最後のフェイズで確定される．このフェイズ q を何回繰り返すか，各繰り返し時点でどの程度詳細な中間オブジェクトを求めるかは，プロジェクト管理者が決定しなければならない．

③ $Wb(q) < W_{min}$ の場合

このシステムは小規模であり，作業の重みが小さい．フェイズを構成する設計作業はプロジェクト管理の対象であり，このままでは管理対象として小さすぎるため個々のフェイズで管理作業を行うと冗長度が高くなる．このため近傍のフェイズをいくつかまとめて管理作業に見合う適切な大きさの作業量とする必要がある．表2では，その例として，関係深い基本フェイズをまとめて『要求分析』フェイズと『ソフトウェア設計』フェイズの二つにしている．

5. フェイズ中心型プロセスの構成法

前節までに，フェイズと各フェイズで確定すべきオブジェクトおよびそれを確定するためのプロセスが決定された．フェイズの中では，実行すべきプロセスを繰り返して目的のオブジェクトを確定し，さらにその裏付けを行うためにプロセスの先読みを行う．フェイズの中のプロセスは，この繰り返しと先読みからなる．以下に各フェイズの中における設計プロセスの実行シークエンス作成について述べる．

5.1 フェイズの中のプロセスの繰り返し

簡単なシステムを設計する場合は，各フェイズ内に含まれるオブジェクト中心型プロセスを1回通して実行するだけでそのオブジェクトを確定することができる．しかし，実用システム設計においては，一つのフェイズで必要なすべての入力オブジェクトが同時に獲得できない場合や，そのフェイズで確定すべきオブジェクトが1回のプロセス実行では完成できない場合があり，そのフェイズの中で同じプロセスを繰り返し，逐次実行して最終的にそのフェイズの目的とするオブジェクトを確定する．この繰り返しの回数は計画段階と実行段階とでは異なる場合がある．このようなプロセスの繰り返しが未経験のシステムを設計する場合に多くなるのは当然であるが，既に経験を積んだシステムを設計する場合でも必要である．

フェイズ p のプロセスは，フェイズ p を構成するプロセスの繰り返し，または，フェイズ p のプロセスとその次のフェイズのプロセスを含む繰り返し実行から成り立っている．

5.2 一つのフェイズの中でのプロセスの先読み

図6には，各フェイズの目的とするオブジェクトを作成するためのプロセスが記述されている．しかし，実際の設計において一つのフェイズの中で目的とするオブジェクトを確定するためには，そのフェイズの中で，本来もっと後のフェイズで実行すべきプロセスを仮に実行し，得られたオブジェクトがそれ以降のプロセスの実行に十分なものであることをチェックしている．

図6の中の第2の『システム仕様設計』フェイズの目的は，2『運用検討』，3『機能検討』，4『システム仕様のフィージビリティ・スタディ』の各グループに属するオブジェクトを確定してシステム仕様を明確にすることである．このシステム仕様は，その後のフェイズ3で設計されるソフトウェアによって実現されなければならない．フェイズ2の中でシステム仕様を確定するに当たり，本来ならばフェイズ3で実行すべきソフトウェア設計プロセスをこのフェイズ2の作業としてあらかじめ実行し，その実現性を評価する．すなわち，E『データおよび処理検討』プロセスによりソフトウェアの構成，データフローダイヤグラムなどを検討し，さらにF『制御およびタイミング検討』プロセスによりデータの流れおよびそのタイミング，制御関係，順序などを検討する．さらにG『ソフトウェア設計のフィージビリティ・スタディ』のプロセスにより，これらの結果を基にしてソフトウェアの実現性をチェックするとともに応答時間性能の近似値等を得る．これをプロセスの先読みと呼ぶ．このプロセスの先読みをどの程度先のプロセスまで行うかは設計者がその経験によって定めることになる．

図7に上記で示した繰り返しと先読みを含む実際のフェイズ中心型プロセスの例として実用システムDASの実際の設計プロセスを示す．各プロセスのブロックの大きさは，作業量をマクロ的に示す．また，斜線を施したプロセスの出力オブジェクトは斜線のプロセスを含むフェイズで確定されることを示し，その主な作業は，斜線のプロセスの時点で実行されることを示す．上の欄には第1レベルのフェイズ中心型プロセス，すなわちフェイズが示されており，左から右に向かって実行される．図中の各フェイズの下には，各フェイズの中の具体的なプロセスが示されている．一つのフェイズの中のプロセスは上から下の順に実行され，それが終了すると右隣の列のプロセスが上から下に向かって実行される．このDAS設計のフェイズ中

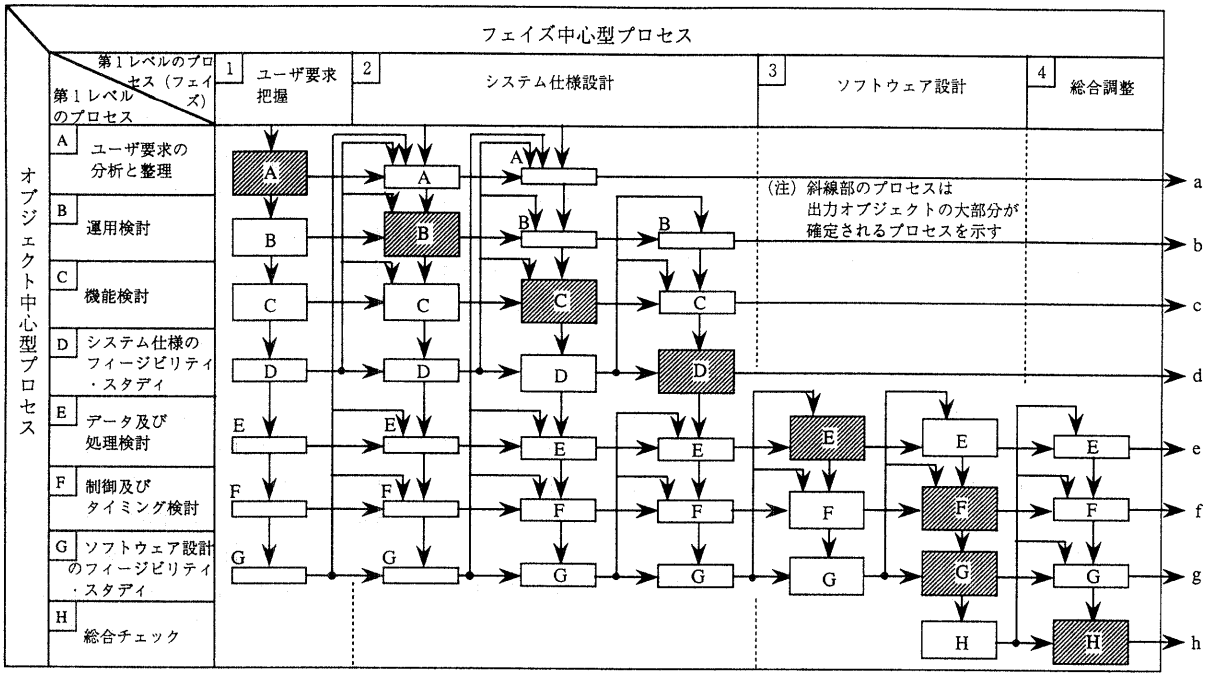


図7 実用システムDASのフェイズ中心型プロセス
Fig. 7 Phase-centered process for designing the practical system DAS.

心型プロセスでは，『システム仕様設計』フェイズの中で繰り返しが3回，『ソフトウェア設計』フェイズで2回行われている．

6. まとめ

われわれは，HFSPのプロセスの定義とその構造に基づいて部門の標準的な設計手順を分析し，記述した．その結果オブジェクト中心型プロセスとフェイズ中心型プロセスという二つのプロセス記述のためのモデルがあることを発見した．熟練技術者は豊富な設計経験の中から普遍的なオブジェクト中心型プロセスを抽出し，実際のシステム設計に当たりこのオブジェクト中心型プロセスをそのシステムに適切なフェイズ中心型プロセスに変換して再利用している．本論文では，これら二つのプロセスの記述方式をオブジェクト遷移を基にしてその構成を明らかにした．さらに一つのシステムの設計で作成されたオブジェクトを分析し，それを基にして具体的なシステムを設計するときのフェイズの構成法を示した．また，そのフェイズの中でオブジェクト中心型プロセスをどのように実行するかを示し，これによってフェイズ中心型プロセスの構成法を示した．この研究によってこれまで曖昧であったフェイズの作成方法を明らかにした．

本構成法は一つにまとまったチームで開発するシステムであれば種類を問わず適用できる．また，大規模なシステムに対しては，サブシステムに分割してサブシステムごとに適用することができる．また，本構成法は，ある程度わかっているシステムの設計に適用できるが，どのようなオブジェクトを作成すべきか具体的にわかっていないシステムに適用することは困難である．

この研究を通して，熟練技術者の実用システムの設計手法を明らかにすることができたが，我々は設計プロセスの記述方法をさらに追求し，将来，プロジェクト管理も包含する方向に発展させたいと考えている．

参考文献

1) Osterweil, L.: Software Processes are Software too, *Proceeding of the Ninth International Conference on Software Engineering,* Montery, California, pp. 2-13 (April 1987).
2) *Proceedings of the 9th International Software Process Workshop* 1～6 (Oct. 1994)
3) *Proceedings of the Third International Conference on the Software Process* 1～6 (Oct. 1994)
4) Watts, S. H. 藤野喜一監訳：ソフトウェアプロセス成熟度の改善，日科技連 (1991)
5) Katayama, T.: A Hierarchical and Functional Software Process Description and Its

Enaction, *Proceeding of the 11th International Conference on Software Engineering,* pp. 343-352 (1989).
6) Mochizuki, S., Yamauchi, A. and Katayama, T.: Analysing and Evaluating Fundamental Design Process of Checkout System for Artificial Spacecraft, *Proceedings of The Fifteenth Annual International Computer Software and Applications Conference (COMPSAC'91),* pp. 507-514 (1991).
7) Frailey, D. J.: Defining a Corporate-wide Software Process, *Proceedings, 1st International Conference on the Software Process,* pp. 113-121 (Oct. 1991).
8) Mochizuki, S., Yamauchi, A. and Katayama, K.: Software Process and Its Reuse in Design Work of Real Time Processing System, *Proceedings, Joint Conference on Software Engineering '92,* pp. 81-88 (Mar. 1992).
9) 望月, 山内, 片山: 人工衛星チェックアウト・システムの基本設計プロセスのプロセス・モデル HFSP による記述とその評価, 情報処理学会論文誌, Vol. 33, No. 5, pp. 691-706 (1992).
10) Mochizuki, S., Yamauchi, A. and Katayama, K.: Two Models for Describing Software Design Process: Object-Centered Model and Phase-Centered Model, *Proceedings, the 5th International Conference on Software Engineering and Knowledge Engineering,* pp. 291-295 (June 1993).

(平成 6 年 3 月 17 日受付)
(平成 7 年 5 月 12 日採録)

望月 純夫 (正会員)

昭和 14 年生. 昭和 38 年東京工業大学理工学部電気工学科卒業. 同 40 年同大学大学院理工学研究科修士課程修了. 同年三菱電機 (株) に入社後, 一貫して実時間処理システムの開発に従事. 同 62 年三菱スペースソフトウェア (株) に移る. 平成 6 年三菱電機 (株) を退職. 現在, 淑徳短期大学教授. システム設計に関する研究, 設計現場におけるソフトウェア設計技術の記述に関する研究を行う. 電子情報通信学会, ACM 各会員.

片山 卓也 (正会員)

1939 年生. 1962 年東京工業大学理工学部電気工学科卒業. 1964 年同大学大学院修士課程修了. 工学博士. 日本アイ・ビー・エム (株), 東京工業大学工学部助手, 同助教授, 同教授を経て, 1991 年より北陸先端科学技術大学院大学教授. この間, オートマトン理論, プログラミング言語, 属性文法, 関数型プログラミング, ソフトウェア工学などに関する研究を行う. 日本ソフトウェア科学会, ACM, IEEE 各会員.