TECHNOLOGY REVIEW

UNISYS 技報 通巻 23

1989年11月発刊 Vol. 9 No. 3

特集：大規模システム開発

日本ユニシス

ISSN 0914-9996

情報化の進展により，企業内および情報処理産業におけるシステム開発は量的に拡大の一途にある．さらに，業務間結合・企業間業務結合が重要視されていることもあり，システム開発は大規模化の傾向にある．吉田伸一は**大規模システム開発の特質とプロジェクト・マネジャの責任**の中で，大規模システム開発の実態と特質について述べ，それらを成功させるために，開発現場で管理者がなすべきことについて提言している．

航空会社間の競争激化は目を見張るものがあり，この競争に勝ち抜くための第一の武器がCRSである．全日空のCRS (ableシステム) は，当社にとって規模，範囲，期間とも前例を見ない大開発であった．中島崟は，**全日空総合旅客システムableのシステム概要と開発マネジメント**の中でableシステムの業務機能，システムの安定運用・大量トランザクション処理等のためのシステム機能について，その概要を紹介している．また長期にわたるシステム開発の経緯と，発生する問題の早期発見とその対応のためのマネジメントについて述べている．

ALMは，経営計画，予算編成・収益管理，原価管理とともに金融機関経営の中核となるものであり，当社ではこれらの業務を総合的に支援する「総合利益計画管理システム (TOPPS)」の開発・提供を行っている．宮田敏光は**ALMシステム (TOPPS/ALM) の開発**の中で，TOPPSを構成するシステム群の一つであるALMシステムの概要とシステム開発の方法，留意点を紹介している．

昨今の金融・証券界における各種システム化動向の中で，最も注目を集めているものに，グローバルトレーディング・システムがある．中尾晴夫は，その一翼を担う**債券トレーディング・システム開発事例**の中で，約11か月間という短期間に開発した約37万ステップの規模のシステム概要および開発のポイントを紹介している．

近年，農協信用事業は他金融機関との競争の激化に伴い，資金量の伸び悩みや運用利回りの低下等，経営環境は厳しい．前山泰治と平嶋孝正は解決策として渉外力・経営管理機能の強化のために開発された**九州地区農協信用事業情報系システム**を一年余りの短期間で開発できた要因を中心に，共同センタとしての情報系システムの概要と仕組み，およびその開発工程について紹介している．

オフィス・コンピュータの使われ方は，経理・給与に代表される業務処理を行う単一オフコン処理から，オンライン・リアルタイム処理のホスト機に使われるまで，その使用形態が多様化している．市川常男は，**オフコンにおける大規模システム開発の事例**の中で，A社の勘定系システムの開発過程を紹介し，さらにオフコンにおける大規模システム開発の方法を述べている．

大規模小売業の情報システム化は統合化の方向にあり，システム規模の拡大と関連分野の広がり・複雑化を招いている．福原俊作は，**大規模小売業の統合情報システム構築事例**の中で，総工数3000人月の開発システムの内容を述べると共に，大規模化しているシステム開発の構築環境整備に焦点を当て，その工夫点等を紹介している．

コンピュータ・システムの先駆けとして，大部分の企業で稼働してきた人事給与システムは，システム化の時期が早かったゆえに種々の問題を抱えたシステムとなり，再構築の対象として取り上げられるようになった．若吉修治は，**K社新人事情報システムの開発**の中で，再構築の過程を述べると共に，その開発に際し適用した開発技法・プロジェクト体制等に言及している．

システム開発に携わる人間を取り巻く環境は，商用コンピュータが普及し始めてから約15年の間に激しく変化した．コンピュータの性能は大幅に向上しており，システム規模も大規模化の一途である．中村祥次郎は**多元的組織環境でのシステム開発におけるマネジメント**の中で，このような状況に耐えうる組織，組織の活動を生産的なものにする組織について考察している．

特集「大規模システム開発」の発刊によせて

早川公正

今，企業では戦略的情報システム（SIS）の開発が要請されている．この必要性が広範囲に認識されてきている．この要請の一環として，部門間横断システム，企業間連結システム，あるいは国際間連結システム等が構築されるケースがふえてきている．一昔前の部分的システム化に比べ，必然的に構築するシステムが大規模化してきている．

構築するシステムの規模が大きくなると，開発に要する費用，開発の難度は，システムの大小との比例関係以上に増大する．さらに，システム内のモジュール間で整合をとる部分が指数関数的にふえる．運用開始後障害が発生した時の影響も大きいため，それを防ぐ，または被害を最小に押さえるために，障害防止機能や障害回復機能に開発者は多大な工夫をほどこす．

大規模システムは一般的にライフサイクルが長い．時間とともにシステムに対するニーズは変化する．そのため長期にわたるニーズを捉える必要が生ずる．それでもシステムの変更は発生する．

したがって，変更の可能性のある部分を予測し，変更しやすいシステム構造を考える必要がある．ライフサイクルが長い分変更に強いシステムを作ることが要請される．

大規模システムは投資額も大きく，納期が遅れた場合，その分費用その他に影響を与える．また，運用開始後の障害時の影響が大きいため品質に対する要請も強い．

大規模システムは「確実に開発する」ということが重要である．「確実に開発する」世界に近づくため，新しい技術・手順・原則・標準が提唱され，実験され，宣伝されてきた．ノウハウも蓄積され，その活用もはかられている．

しかし，それだけで大規模システム開発が成功するものではない．ここに大規模システムの開発を担当，あるいは任された人達の苦心・苦労がある．

大規模システム開発に対するお客様の要求は，最近ますます増加している．

日本ユニシスは，お客様の要求と期待を受け，お客様の真のニーズをシステム化することが最大の使命であると認識している．

本号では大規模システム開発の実態・特質を述べ，開発現場で管理者がなすべきことを提言するとともに，金融・証券・運輸・製造・流通の幅広い業種アプリケーションから大規模システム開発分野の事例の一端を紹介し，あわせて知識労働者から構成される多元的組織環境での大規模システム開発におけるマネジメントはどうあるべきかについて述べた論文を最後に掲載している．

今後ますます重要になってくる「大規模システムを確実に開発する」という課題を自らの課題とされる多くの人達にとって，本号が少しでもお役に立つことを期待したい．

（システム技術本部　本部長）

大規模システム開発の特質とプロジェクト・マネジャの責任

Characteristics of Large-scale Systems Development and a Project Manager's Responsibility

吉田伸一

要　約　情報化の進展により，企業内並びに情報処理産業で実施されている情報システム開発は量的に拡大の一途である．そして近年，業務間結合・企業間業務結合が重要視されていることもあり，システム開発の大規模化も今後の傾向となっていくであろう．

本稿では，開発作業の自動化が序々に進んでいるものの依然としてシステム開発，なかでも大規模開発の前に横たわっている解決しがたい問題があることを述べ，大規模開発を成功裏に終わらせるためには，開発の現場で日々指揮をとるプロジェクト・マネジャが，まずこれらの問題を認識しなければいけないこと，次にこの認識のもとに最低限全とうすべき責任があるということを述べた．

多くのプロジェクト・マネジャが，失敗の経験により成長していくということでは必ずしもなく，本稿のような紙面を通して，大規模開発を成功に導くヒントを得られるようにすることが本稿の目的である．

Abstract　As information society advances, efforts to develop information systems both in businesses and in the information processing industry have continuously been on the increase in volume. Linkage between computer applications and application linkage between businesses have come to be regarded as more important in recent years, and so the future trends would also move toward expansion in the scale of systems development.

This paper refers to the fact (1) that there are still hard-to-solve problems lying before systems development, particularly large-scale systems construction projects, (2) that success in such development projects requires a project manager who acts daily as overseer at the workshop to recognize those problems before anything else, and (3) that there is a range of responsibility he has to cover at the least based on his own recognition.

Project managers do not always grow through their experience in failing. This paper is intended to provide them with some hints which would lead to successful large-scale systems development.

1. はじめに

日本における情報化の進展は一貫して上昇傾向にある．表1[1]で見る通り，対GNP比によるコンピュータの出荷額は1987年に1％を超えた．これに関連し表2[1]を見ると，情報サービス業の売り上げはサービス業務すべてで上昇しており，とくにソフトウェアサービス業での伸び率が高い．この点アメリカにおいても同じ傾向にあり，ソフトウェア関連産業の堅調さがうかがい知れる．

上記のように，情報処理産業並びに企業内で実施されている情報処理システム開発(以下システム開発と記す)は量的に拡大の一途であるといえる．この中で，大規模システム開発に該当するものは，表3[2]で推測するとシステム開発総コストの60％を超

表1 コンピュータ出荷額対GNP比各国比較[1] (情報化白書1989, p. 83から)

Table 1 The amount of shipment-based money v. s. GNP
——several countries comparison

国＼年	1983	1984	1985	1986	1987
日本 (1)	23,402 (0.834)	25,137 (0.842)	29,079 (0.916)	32,746 (0.988)	35,615 (1.032)
アメリカ (2)	419 (1.231)	535 (1.418)	553 (1.377)	525 (1,238)	535 (1,182)
イギリス (3)	1,543 (0.512)	2,471 (0.764)	3,807 (1.076)	3,733 (0.985)	4,025 (0.972)
西ドイツ (4)	103.3 (0.614)	132.7 (0.750)	164.8 (0.893)	166.6 (0.856)	176.2 (0.872)
フランス (5)	236.0 (0.589)	288.5 (0.661)	331.9 (0.706)	349.3 (0.694)	367.7 (0.696)

上段はコンピュータ出荷額, 下段( )内は, 対GNP百分率 (%)

〈資料〉(1) 通商産業省「電子計算機納入下取調査」,
電子協「オフコン・ミニコン・パソコン出荷状況調査」(単位：億円)
(2) アメリカ商務省 Industrial Outlook (単位：億ドル)
(3) Machintosh (単位：百万ポンド)
(4) Machintosh (単位：億マルク)
(5) Machintosh (単位：億フラン)

表2 情報サービス売り上げ対GNP比の日米比較[1] (情報化白書1989, p. 83から)

Table 2 The information service proceeds v. s. GNP
——Japanese/U. S. A comparison

国	内訳＼年	1983	1984	1985	1986	1987
日本	情報処理サービス業 (1)	5,343 (0.190)	6,308 (0.211)	6,372 (0.200)	7,045 (0.212)	7,582 (0.219)
	ソフトウェアサービス業 (1)	3,643 (0.129)	5,123 (0.171)	6,580 (0.207)	9,127 (0.275)	11,045 (0.320)
	情報提供サービス業 (1)	787 (0.028)	966 (0.032)	1,007 (0.031)	1,143 (0.034)	———
	データベースサービス業 (1)	———	———	———	———	432* (0.012)
	合計	9,773 (0.348)	12,397 (0.415)	13,959 (0.439)	17,315 (0.522)	19,059 (0.552)
アメリカ	情報処理サービス業 (2)	128 (0.375)	148 (0.392)	163 (0.406)	186 (0.439)	217 (0.479)
	ソフトウェアサービス業 (2)	200 (0.587)	255 (0.676)	300 (0.747)	373 (0.880)	457 (1.009)
	情報提供サービス業 (3)	19 (0.055)	23 (0.060)	26 (0.065)	23 (0.054)	27 (0.060)
	合計	347 (1.019)	426 (1.129)	489 (1.218)	582 (1.373)	701 (1.548)

上段は売上額, 下段( )内は, 対GNP百分率 (%)

〈資料〉(1) 通商産業省「特定サービス産業実態調査」(単位：億円)
*;「情報提供サービス」と「データベースサービス」については, 調査項目の変更により, 年次間の比較はできない。
(2) INPUT Information Services Industry Report (単位：億ドル)
(3) CSP International, SRI (単位：億ドル)

表3　ソフトウェア開発の対象コンピュータ別売上高[2]（情報サービス産業白書 1989, p. 84 から）
Table 3　The software development proceeds distinguished into the computer system scale

（単位：%）

	汎用機				会計処理用オフィス・コンピュータ	ミニコンピュータ	パーソナル・コンピュータ	計
	大型	中型	小型	超小型オフコン				
情報処理サービス業	43,683 (84.4)	3,483 (6.7)	1,202 (2.3)	1,412 (2.7)	29 (0.1)	891 (1.7)	1,032 (2.0)	51,732 (100.0)
ソフトウェア業	172,578 (56.0)	35,645 (11.6)	30,660 (10.0)	10,797 (3.5)	2,033 (0.7)	36,434 (11.8)	19,909 (6.5)	308,056 (100.0)

有効回答数―情報処理サービス業58
―ソフトウェア業143

えていると思われる．ただし，件数的には全体の 10 %以下であろう．近年，業務間結合・企業間業務結合をもとに組織体の競争戦略を具現化する‘戦略的情報システム (SIS : Strategic Information System)’が登場してきたこともあり，システム開発の大規模化も今後の傾向として続いていくと考える．

本稿では，大規模システム開発の実態と特質について述べた上で，それらを成功裏に終わらせるために開発の現場で日々指揮をとる責任を負った実質的管理者 (Project Manager，以下 PM と記す) に焦点を絞り，彼等がなすべきことにつき提言する．

2. 大規模システム開発の実態と特質

2.1 大規模の定義

NASA のシャトル計画での宇宙船用ソフトウェアの開発規模は，およそ 5000 万行 (アセンブラ言語と高水準言語を併用) といわれている[3]．これは明きらかに大規模である．ところでシステム開発の規模の定義をすることは，これまであまり興味を持たれなかったといえる．高水準言語や開発支援ツールの活用や PM の経験度により同じ何十万行といっても，意味あいは変化していくし，独立性の高いサブシステムを複数個同時に開発した場合，全体は大規模開発でも個々のサブシステムでは実質小規模開発でしかない場合もあろう．

しかしここでは，大規模システム開発に定量的なイメージを与えるため，R. N. シャレット氏の数字[3]を参考にして大規模システム開発を次のように定義しておく．

> 『一つの開発テーマに属し，並行に進行し一定期間内に終了する関連性の高い部分開発の集まりが，言語によらず原始行で 10 万行以上になった，もしくはなると予想されるとき，その部分開発の集まり，または開発全体を大規模システム開発と呼ぶ．』

当社が年間で着手するシステム開発は約 3000 件で，このうち大規模といえるものは約 5～6 %である．これまで，大規模システム開発といえば汎用大型コンピュータ用がほとんどであったが，今後はオフィスコンピュータやそれらと汎用コンピュータとの複合システム用の大規模開発の増加も予想され，大規模システム開発の中身は，より多様化・複合化していくことになるであろう．

2.2 システム開発は何を持って成功とするか

次から次へと開始し終了していくシステム開発が，成功したのか失敗したのかを判断する尺度として，'納期' '費用' 機能と安定性とを合わせた '品質' の三つがよく使われる．この三つの尺度で，成功とはどういう状況かを考えてみると，三点とも実績が予定より良ければ，もちろんシステム開発は成功したといってよいであろう．問題はそれ以外の場合である．一点でも実績が予定より悪ければ '失敗' と判断する考え方もあるが，そもそも予定の設定の仕方が多分に感覚的・条件付きであることを思うと，これは現実的ではない．そして納期・費用・品質共に当初の予定値を満たすということは，大規模システム開発では非常に少ないと考えるから，ほとんどの大規模開発は '失敗' と判断されることになる．

開発の目的が「とにかく稼働開始すればよい」という極端なことになっていたとすると，三点とも失格でも，システムが動き出せば結局は成功したと言えるかもしれない．

以上により，'システム開発成功' の定義を次のようにして論を進める．

『システム開発の企画の承認者が納期・費用・品質の三つの尺度のうち最優先と決めた尺度において，実績が予定より良く，他の尺度についても，第三者の目で妥当と評価されるとき，該システム開発は成功とする．』

この定義をなぜ必要とするかについては，以下大規模システム開発の実態と特質につき順に述べた後で説明したい．

2.3 筆者の経験

筆者が10数年前にPMとして初めて手掛けたシステム開発は，原始行で4万行弱であり，いわば中規模開発であった．納期は守ったものの，予定費用をかなり超過し，計画した機能の約3分の1を先送りすることになってしまった．

システムは稼働し始めたが，成功の定義からすると失敗であった．原因は今から思えば単純であり，ソフトウェア工学とかプロジェクト・マネジメントとかが，叫ばれる前であったので筆者自身もシステム開発の手順や手続，管理項目につき，十分理解せずに千行のプログラムを十数本作ればよいという程度の認識で作業を始めたことにあった．

本稿では，システム開発で基本となる手順・手続・管理項目については説明しないが，システム開発は 'これらの基本' を知って実施しないと失敗に至ることを実証してしまった訳である．これを契機に自己流ではあったが，システム開発の基礎が急速に身につき育っていった．

筆者の例のように，'失敗は成功のもと' といったPMの成長のさせ方が一番良いのかも知れないが，大量のシステム開発をかかえる組織体においては，失敗による損失も多大となる．多数のPMが，最少の経験のみで大規模システム開発に挑戦し，そして成功する方法が今や求められていることであろう．この課題に少しでも役立てることを願って本稿を進めたい．

2.4 大規模と中小規模の分岐点の意味

プログラムの原始行の量でシステム開発の規模を定義したが，ここで大規模特有な状況とはどういうことかを述べる．まず当然ながら，システム開発において10万行を

超えると大きな状況変化が突然発生するということではない．

大規模開発では，場所の問題で分散開発をせざるを得ず，その管理方法が問題になってくるということは確かにあるが，そういう固有の問題を論ずる前に「基本として守るべきこと，やるべきことを十分理解した上で，それらを確実に実施していかないとシステム開発が失敗に至る分岐点が，およそ10万行のあたりにある」と考える．言葉を変えると，「システム開発の基本動作は大は小を兼ね，まず大規模開発用にあり，規模の縮少に従って部分的に省略することが可能になる」といえる．

2.5 解決しがたい問題

要求定義からシステムテスト終了に至るシステム開発工程（図1）全般にわたり，その自動化には限界がある．いかにソフトウェア工学や人工知能技術が発展しても，そうであろう．そして，この限界を超えた部分は人的要因に頼らざるを得ないが，そこを中心にいくつかの解決しがたい問題，大規模開発での実態そのものといえる問題が横たわっている．以下にこれらを列挙する．

2.5.1 企 画 段 階

1) 要求定義……システムの利用者は，該システムで実現したい機能をまず明らかにしなければならないが，この要求定義がこの時期になかなかまとまらない．

大規模開発では，開発が終了した時に要求定義がやっと完成するとまで言われている．

2) 見積り……要求定義のまとまり具合とも関連するが，開発の規模，要員の必要工数，期間，費用の見積りを，開発を始める前に正確に予測できない．ほとんど

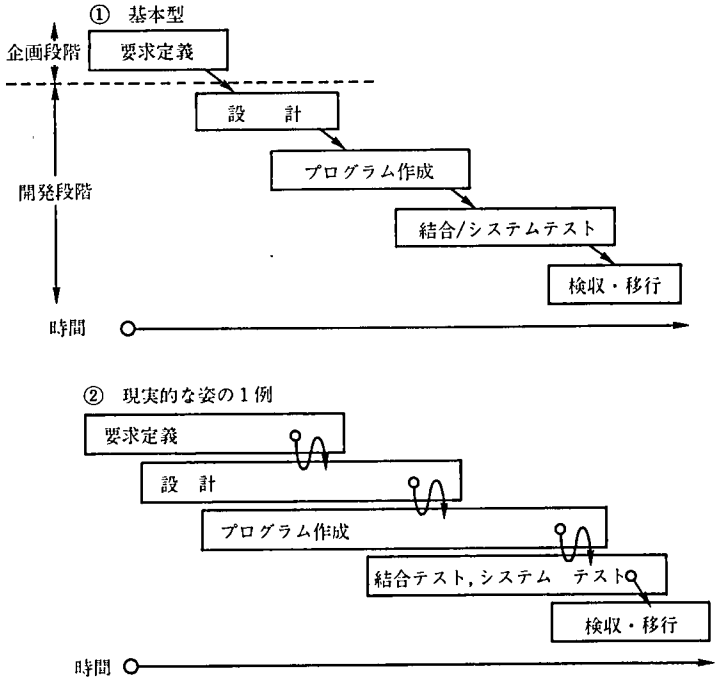


図1 ウォータフォール・モデル

Fig. 1 A water fall model

が過少予想になってしまう．もしくは，それらの一部が企画の承認者より前提として与えられてしまう．

3) 体制……体制，とくに質的な面が整わないまま企画段階をスタートせざるを得ない状況がしばしば発生する．

4) 新技術の採用……新しいハードウェアやソフトウェアの新技術を採用することになった場合，それらの実現性や利用方法をチェックできないまま企画や開発をスタートせざるを得ない状況が発生しやすい．

2.5.2 開 発 段 階

1) 体制……企画段階で計画した量と質を満たす開発体制を，当初から整えることがなかなかできない．さらにシステム開発未経験者を含む多数のメンバについて能力や適性を十分に把握することができないままで，役割分担せざるを得ない状況になりがちである．

2) 設計の手戻り……システム開発全般にわたり，作業の手戻りがいろいろ発生する．なかでも要求定義の不正確さ不十分さや設計自体の問題により，設計の手戻りが多発した場合，開発の工程管理が混乱する一方，それまで作成してきたドキュメント類の更新が十分行われなくなってしまい，後々プログラム間の不整合問題を発生させてしまう．

3) 開発作業管理……システム開発の工程モデルとして，伝統的に用いてきた'ウォータフォール・モデル'を，大規模開発では工程間の後戻りが頻発しがちなため不向き[3]とわかっていながらも，'理想的なモデル'が未完成なため，結果として採用せざるを得ない．このため，工程モデルはミクロの世界では十分には役に立たず，PMは日常の進捗管理や作業分担の変更管理等をかなり負荷をかけて実施しなければならない．そして，この努力を放棄すると即，開発プロジェクトは管理者不在の状況に落ち入ってしまう．

4) テスト項目……一つのデータに対し判断条件が，たとえば15個あると最大32,768ケースのテストが必要になってしまう．テスト技法はいくつかあるものの必要十分なテスト項目を列挙するということは，一般に非常にむずかしい．

5) 外部委託の管理……システム開発の部分を外部委託しているとき，開発の多忙さに追われ，もしくは会社が違うことでの遠慮によりPMの目がその部分からつい離れてしまい，手遅れの状況で外部委託分の中の問題が発覚したりする．

2.6 大規模システム開発の特質

大規模システム開発の定義や実態につきこれまで述べてきたが，ここで昨今の経済的・社会的動向を踏まえて，大規模システム開発の特質をまとめてみると以下の三点となる．

2.6.1 経済的・社会的役割の重要性

平成元年度の経済白書によると，1986年12月以来の日本の大型景気を推進する経済の大きな構造的変化（生活と産業の高度化・グローバル化の進展・資産残高の増加）を作り出し，日本の産業界を高度にし，強い競争力を生んだ大きな要因は情報化の進展にあるという[4]．この情報化の中核をなすのが，次々と開発され運用されている大規模情報システムであるといえる．

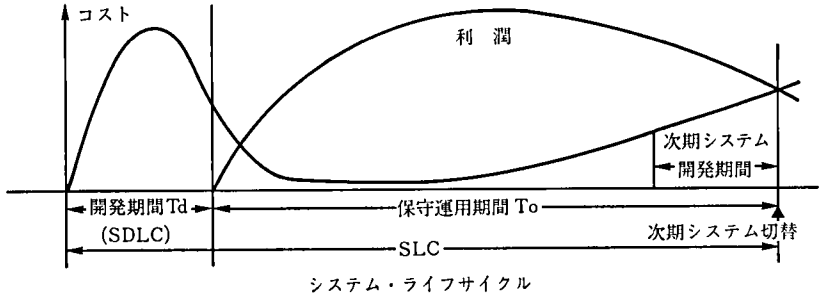
したがって，大規模システムは経済的・社会的役割の重要性の認識のもとに開発していく必要があるが，同時にその脆弱性の認識も合わせて持っていなくてはならない．すなわち，情報化の進展に比例して，情報システムの停止・悪用・有効に機能しなくなることの企業等の経営活動への支障，国民生活全般への影響は，より深刻になっていく．これを防ぎ，情報システムの健全化を図るための考慮を，システムの企画段階より，業務処理への利用者の機能要求の検討と並行して，十分実施しておかなくてはならない．この健全化を図るための一つの有効な手段として，第三者の調査と助言・勧告を得る‘システム監査’があり金融機関中心に活用されている．

今後は監査といっても，情報システムが運用段階に入ってから評価するというのではなく，企画の当初より運用段階に至るまで複数回の過程監査を実施するという考え方に立つ必要がある．

2.6.2　システム・ライフサイクルの考慮の必要性

システム開発の大規模化は開発期間の長期化とほぼ同義ともいえる．桃山学院の井上義祐助教授が規定しているシステム・ライフサイクル(‘開発期間’+‘保守運用期間’)について考えてみると，開発期間が長期化すると該システムで十分な利潤をあげるためには，保守運用期間をかなり長期間持ちこたえさせなければならない．井上助教授の経験では保守運用期間は開発期間の3倍以上必要とのことである（図2）[5]．

日本経済の新しい潮流は，変化への迅速な対応も求めているので，元をとってから次期システムの検討に入るという図式は，一般的には当てはまらない．必然的に開発期間の短縮が常に要求されることになる．このことは，システム開発を承認する経営者にとって，システム・ライフサイクルを決めた上で投資効果を算出する必要があることを意味している．そして，設定されたライフサイクルを満たすために経営者，利用者，PMが一体となって，2.2節で述べた納期・費用・品質の予定につき，最大限検討し決定をしなければならない．



$$\left[\begin{array}{l}\text{SLC の経済的効率係数(Ee)} = \dfrac{\text{SLC 期間中の利潤}}{\text{SLC 期間中の出費}} \\ \text{SLC の期間的効率係数(Et)} = \dfrac{\text{運用保守期間(To)}}{\text{開発期間　(Td)}}\end{array}\right]$$

図2　SLCにおける期間的効率係数概念（ET）[5]

（システム監査学会第3回研究大会の報告要旨，
情報システムの新展開とシステム監査 p. 29 から）

Fig. 2　A concept about the system life cycle

2.6.3 PMの責任の重大さ

システム開発の失敗例の分析は実施しずらく，その結果は公表しずらいものである．2.3節で述べた筆者の例でも，冷静に自己批判できたのは，開発終了後数年経てからであった．この辺の事情を読者にご理解いただいた上で，システム開発の失敗例の分析結果について総括してみると，失敗原因のほとんどが，2.5節に記した解決しがたい問題を認識せずに，もしくは認識していても妥当と思われる対策を打てずに企画・開発を進めたことにある．

その原因はPMの管理の方法と内容のまずさにあるといわざるを得ない．繰り返しになるが，大規模システム開発はそのスケールのゆえに解決しがたい問題が横たわっている．しかし，これらは，最善の対策を取ることはむずかしいが，次善の策，妥当な策で，損失を最少にするということで解決しなければならない問題である．

PMがこの点を認識し努力しなければ，大規模開発は成功することはないであろう．これだけPMの責任は重大である．一方，開発をPMに依託する経営者・上位管理者は，三つの尺度（納期・費用・品質）をすべて満たすことをPMに求めるのは当然ではあるが，場合によっては優先度を決めてそれを明快にPMに示し，トレードオフのための根拠を与えてPMを支援すべきである．

3. 大規模システム開発におけるPMの責任

アメリカ経営者協会は，マネジメントの定義を「他の人を通じて仕事を成し遂げること」としているが[6]，これを表面的にとらえると，どんな問題が発生しようと何人かのメンバを集めて「早急に君達で対策を立てて解決を図りなさい」と命令だけしていればよいということになる．

もちろん，これでは大規模開発のPMは勤まらない．日頃の統制を確実に行い，問題の早期発見と把握に努め，発生した問題によっては率先して解決に当たらなければ，他の人を通じて仕事を成し遂げることにはならない．

本章では，経営者および上位管理者の責任についてふれた後，大規模システム開発におけるPMの基本的責任事項について述べる．

3.1 経営者および上位管理者の責任

開発組織を直接指揮はしないが，システムの企画の承認をする経営者や管理者，そして開発プロジェクトのPMの上位管理者の責任について述べる．

1) システム・ライフサイクルの決定……まずシステムの開発期間と保守運用期間を決めて，納期・費用・品質の予定をPMが十分納得できる根拠を持って定める．そして，この三つについてはすでに述べたように，場合によっては優先順位を決めなくてはいけない．納期や費用優先ならば，他企業との競合上多少問題があっても，「この機能は当社では当面具備しなくてもよい」といった機能縮小判断がたぶん必要になる．品質優先ならば，納期や費用についてはPMに予定を提案させた上で決めたり，時には「該情報システム自体は直接利潤を生まなくてもよい」といった組織体としての総合的な判断も必要になる．

2) PMの選任と開発体制……開発プロジェクトの組織化は重要な責任であり，この中でPMの選任は，配役を間違うと多大な損失を生む可能性があるのでとくに

重要である．このため，OJT や OffJT により信頼できる PM の育成体制を確立しておく必要があり，その内容は当初から大規模開発で発生する問題点と対応例につき，実践的に教育するものでなければいけない．

3) 第三者による監査……開発プロジェクトを上位管理者が適宜レビュすることはそれ自体重要であるが，経済的・社会的もしくは国民生活に重要な関わりを持つシステム開発においては，第三者すなわち社内外のシステム監査人による監査を実施することを今後は必須としたい．この監査人は，コンピュータに詳しく PM 以上の見識を持っていないとおそらく勤まらないので，組織体としては PM と共に優秀な監査人の育成についても計画的に進めていかなければならない．

3.2 PM の責任

システム開発プロジェクトを日々直接管理し，予定の納期・費用・品質を保ち，開発を完了させるのはまさに PM であるが，この PM の責任すなわち守るべきこと，やるべきことの主要なものにつき次に述べる．

1) 基礎教育の受講

大規模開発の手順や手続，支援ツール，さらにその実態や特質につき，一通りの基礎教育を受け，理解することが必要である．過去の事例を学び，一定期間 PM の補助者として先輩の行動から学ぶことができれば，より実践的に理解できるようになるであろう．

2) 企画段階

① 要求定義の明確化：何を作るかを明確にする要求定義は，システムの利用者の責任のもとで作成されていなければならない．この要求定義の存在をまず確認する．存在していないか，またはそれが不十分な場合は利用者に協力し十分な要求定義を作成させることが PM の最初の仕事になる．ポイントは要求定義の内容を PM 自身が納得することにある．納得するということは，次に実施しなければならない見積り，すなわち開発の規模，要員の必要工数・期間・費用の見積りが，ある前提のもとで自信を持って実施できる程度に納得できればよい．納得できない時は，開発段階に進んではいけない．また納得できたとしても，開発段階に入ってから追加要求が出されると思われる部分が残っていれば，その程度によって進むか停まるかを判断しなければならない（図 3）．

② 適切な見積り：見積りはむずかしい．大規模開発では予定が 100 万行のとき，5％の誤差では中規模システム 1 個，10％の誤差では大規模システムをもう 1 個，並行して開発してしまうことを意味する．しかし，通らねばならない関門であり訓練により精度は向上していくものである．ポイントは，前提を常に明確にしておくことと，類似システムの開発経験者の意見を納得した上で取り入れることである．見積り技法も有効である．また要求定義のまとまり具合によっては，開発段階の初期にもう一度やり直して精度を上げることも考える．

③ 適所配置と専門性志向による体制作り：PM を選任し，組織の骨組を作り必要な要員を揃えるのは上位管理者の責任であるが，組織の中の階層作りや各メンバの役割り分担を決めるのは PM である．このためには，メンバ 1 人 1 人の能力と適性を面接や簡単なケーススタディ等により十分に把握することを，最

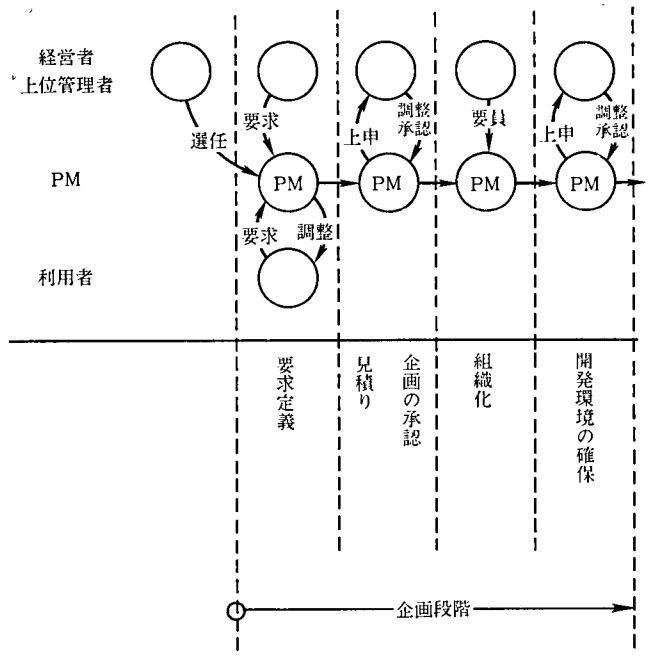


図3 企画段階の中の処理フローの例

Fig. 3 An example of the process flow in a software development planning phase

初に行わなければならない．そして彼等を適所に配置することがより成功に導びく方法と考える．

たとえば，設計者が10名，プログラマが40名必要なところ，設計適任者が6名，プログラマ適任者が44名与えられたのであれば，プログラマ適任者から4名をOJTによる育成前提で設計者に当てることが第1である．適任者がいないと判断される時は，むしろ与えられたままの状態に役割を決め，ずれの問題は工程管理の中で解決の努力をするという考え方の方が良いと思う．もちろん，このずれがひどすぎる時は，開発開始そのものを一考しなければならない．

もう一つのポイントはPMを補佐し，日頃の進捗管理が予算管理，要員管理，各種ドキュメントの管理，開発環境の整備を専門とする小組織を必ず持つべき，という点である．それだけ大規模開発の中で発生し処理すべき，'事務量'が多いということであり，このPM支援組織こそ第1に作るべきものと言ってもよい．そして，全体として専門性志向の方が生産性向上につながると考える．

④ 開発環境の確保：大規模開発では，開発環境(使用コンピュータや端末機器・作業場所・会議室・事務用品等）を企画段階で確保することが必須である．必要なものをすべて上申し，確保する一方，作業場所が複数の分散開発となった場合，分散開発の実情やノウハウを十分収集し，自分達の環境に合った管理方法をここで決めておく必要がある．

3) 開発段階

① 開発方法・技法の設定と遵守：開発方法（手順，手続，作成ドキュメント，

プロジェクト管理方法等）と開発作業を直接支援するために活用する技法（プログラミング技法・テスト技法等）をまず定めることである．そして開発終了まで，確実にメンバに守らせるということが，開発段階での最重要ポイントである．'守らせる'と絶対成功するとは必ずしも言えないが，この逆，すなわち成功した開発のPMは，まちがいなく'守らせている'ため，このことは必要条件といえる．また何を採用するかということではなく，一度決めた規則を守らせることの方が重要とも言える．ただし，組織体全体での生産性向上を図るためには，より標準的な技法を採用し，その技法を用いて得た先輩のノウハウを蓄積し活用した方が望ましいことはもちろんである．

② 開発方法・技法の要点：開発方法・技法の全体像がどうなっているかについて説明することは紙面の都合でできない．読者が理解していることを前提とするが，以下PMの責任で遵守させる開発方法・技法の中でのいくつかの要点を列挙する．

ⓐ 各工程の事前環境整備と役割分担の明確化

工程モデルの問題については先に述べたが，現在はウォータフォール・モデルを基本にして，開発環境に合わせて改良するなり，問題点を工程管理の中で個々に解決していくことになる．工程管理上まず重要なのは，工程ごとの環境整備（ハードウェア，ソフトウェアの準備，ワークシート類の記述練習，作業場所の確認他）を前工程の中で次工程が始まる前までに終了させておくことである．次は工程ごとに，利用者・開発担当者（社内とソフトウェアハウス）間の役割区分を同様に前もって細かく定め，工程ごとの適切な要員の配置と作業スケジュールとを確実に行うことである．

ⓑ 日報による管理の実施

開発担当者が多い大規模開発では，つい週間・月間ベースの管理になってしまう．これは管理側の負担を考えてのことであろうが，この周期での管理は調整の遅れが発生しやすく，修羅場を作りやすい．大規模開発では，日報により進捗状況とメンバの実質的能力を把握しておき，それをもとに細かな進捗状況の報告や，適切な要員再配置がいつでもできるようにしておかねばならない．このため，管理側のPMは該プロジェクトに専任でなくてはいけないし，PMを支援する専任組織が必須になる．

ⓒ プロジェクト内コミュニケーション

PMは，開発組織の中のキーマンには，たえず全体の進捗状況を，知らしめるとともに開発メンバの苦情が彼らから，すぐに伝わってくるための方策を講じ，プロジェクト内の良好なコミュニケーションの維持に努めなければいけない．

ⓓ 設計工程内での機能レベルのテスト仕様作成

最近その実践例をよく聞くが，「設計工程の中で，機能レベルのテスト仕様書を作成すること」が次に示す理由で大へん有効である．

—要求定義と設計のずれをテスト仕様という見方で発見できる．

—テスト不可能な機能が確認でき，その対策を早目に検討できる．

—テストのための必要条件がわかり，環境整備を早目に着手できる．

ⓔ ソフトウェアによるプログラム・ライブラリの管理

設計の手戻りやバグにより関連プログラムの修正を正確に実施するには，全プログラムのソースコードと修正コード，さらにJCL，テストデータを専用管理ソフトウェアにより管理していくことが必須といえる．また，修正コードについては修正の履歴を修正と同時に取るようにできれば，修正内容分析や修正予防になり有益となる．

ⓕ テストの担当者に設計担当者の一部を当てる

先に要員の配置は専門性志向の方が良いと書いたが，テスト担当者については設計担当者を参画させておく方が，プログラミングミスの大きなものを早目に発見することに役立つ．

ⓖ 外部委託の管理

開発作業の一部やある工程をすべて外部委託することは，現在では常識に近い事項であるが，これにまつわる問題も多い．「会社が違うから，いろいろ気配りをして責任を全うしてくれるだろう．作業を一括発注したのだから管理も含めて，まかせないとおかしい」ということで，外部委託分については，PMが管理を放棄しがちになるが，これは大きな誤りといえる．大規模開発では問題発見が遅れると，即数か月単位の納期遅れにつながってしまう．PMは外部委託分についても‘管理’を実施することが原則である．必須と考える点は次の通りである．

—採用した開発方法・技法を遵守させること．

—個々の成果物（ドキュメントとプログラム）ベースによる進捗管理と検証をし，問題点につき委託先リーダと打ち合わせて早期解決を図ること．

—外部委託先の不満事項の把握と対応．

—分散開発がなされている時は，PMもしくは設計のキーマンが定期訪問し，進捗の実態を知るようにすること．

3.3 将来への課題

これまで，大規模システム開発の実態と物質，そして管理者とくにPMの責任につき述べてきた．大規模開発の現場に横たわる解決しがたい問題についても，筆者自身の経験やいわゆる成功を納めたPMの足跡をたどり，いくつか解決策を述べてきた．大規模システム開発担当の際に，基本的に守るべきこと，やるべきことを多少はご理解いただけたことと考える．

ところで大規模システム開発遂行上の問題は，人的要因に頼っている限り問題のまま残っていくであろう．

本稿で述べた中では，

1) 要求定義を文字通り企画段階で完結させる方法，これができないとき，追加要求や変更要求が発生しても最少の人手で発生時点までに作られた成果物に反映する方法，

2) ウォータフォール・モデルに代わる大規模システム開発用に適合した新しい工程管理モデルのあり方，

以上の二つが一番大きな課題であろうと思う．

他にシステムの保守段階まで広げると，保守用のドキュメントの作り方はどうあるべきかという問題が出てくる．システムのライフサイクルの中での保守運用期間を長くとるために第1に，保守が大量になっても，それらを十分吸収し，システムの作り直しをほとんど必要としないプログラム構造のあり方という問題がある．

より現実的な問題として，保守項目が発生した時に，該開発には参加していない第三者が迅速に保守の箇所と方法を知るためのドキュメントのあり方・作り方をどうするか，という問題がある．開発時に作成されていくドキュメントは，必ずしも保守フェーズでは役に立つものでないことは良く知られているし，変更事項を反映していなければ，信頼できるドキュメントではなくなってしまい，プログラムのソースコードそのものが唯一のドキュメントという結果になることも多い．

大小合わせて将来への課題は多く，ソフトウェア工学やハードウェア機器の革新を待つ問題もあるが，これらの解決はソフトウェア工学を研究している人達だけで図るものではない．システム開発，なかでも大規模システム開発を担当したり，支援したりする開発現場の人達が苦しみの中からアイデアを出し，実践で試行することにより，さらに現実に適合する解決等を生み出していくべきものであると考える．

4. おわりに

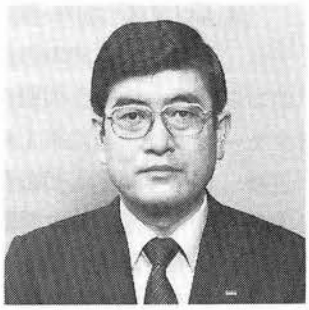
本稿では，大規模開発を担当するPMがまず認識すべきは，解決しがたい問題の存在と大規模開発の特質であり，この認識のもとに，上位の管理者やPMが自覚し全うすべき責任につき述べてきた．

大規模開発では，楽をして成功に至るという王道はなく，PMの管理・統制のもと，開発プロジェクト全体が地道な努力を積み重ねて，守るべきこと，やるべきことを確実に実施しつつ，要所要所で工夫を施こしていくことが，規模が大きいゆえに成果も大きく出て，成功につながるのであると言える．PMを担当する皆さんが，大規模システム開発のむずかしさ・苦しみを理解され，その上で成功へのいくつかのヒントを得ていただければ大へん幸いである．

参考文献
[1] 日本情報処理開発協会，情報化白書1989，コンピュータ・エージ社，1989.5.25.
[2] 通産省機械情報産業局，情報サービス産業白書1989，コンピュータ・エージ社，1989.4.1.
[3] R. N. Charette，(浅岡伴夫監訳)，管理職のためのソフトウェア開発戦略，日経BP社，1988.7.
[4] 朝日新聞，好景気推進する構造変化，1989.8.8，夕刊4面.
[5] 井上義祐，システムライフサイクルの期間的効率係数概念に基づく効率性監査について，システム監査学会第3回研究大会の報告要旨より，1989.5.26.
[6] L. A. Appley，マルチメディア教育システム管理者養成コース，アメリカ経営者協会，(株)国際経営センタ，1971.
[7] 椎羅寛翠，シーラカンスの詩，日本経営科学研究所，Computer Report 1987/10～1989/8.

執筆者紹介 吉 田 伸 一 (Shinichi Yoshida)

昭和42年北海道大学理学部卒業. 同年日本ユニシス(株)入社. 一般企業のSEサービスを担当後, 1100/2200シリーズ運用支援ソフトウェア (UOF) の企画, 開発に従事. 現在, システム技術本部システム監査室長. システム監査学会会員. 内部監査士.



全日空総合旅客システム able の システム概要と開発マネジメント

An Overview of the 'able' System— ANA's Integrated Passenger System and its Development Management

中　島　　崟

要　約　航空業界における各航空会社間の競争激化は目を見張るものがあり，この競争に勝ち抜くための第一の「武器」がCRS(コンピュータ予約システム)である．全日空のCRS，すなわちableシステムは，3年間にわたる約7,000人月の開発を終え，平成元年6月1日に無事カットオーバをしたわけであるが，当社にとって規模・範囲・期間とも前例を見ない大開発となった．

大規模システム開発の事例紹介として，このableシステムがどのような業務機能を持ち，システムの安定運用，大量のトランザクション処理等のためにどんなシステム機能を有しているか，その概要を記述する．また，長期にわたるシステム開発の経緯と，次々に発生する問題の早期発見とその対応のためどのようなマネジメントを行ったか，そのポイントについて報告する．

Abstract Business competition in the airline industry has been markedly more and more intense in recent years. The first 'weapon' which enables airline companies tio survive this strict competition is what is called a computer reservation system [CRS]. The development of All Nippon Airways' CRS—dubbed the 'able' system—which had required about 7,000-man/month workforce for three years running was successfully finalized for its systems cutover on June 1, 1989. This development project turned out unprecedentedly big to NUL in terms of scale/size, application coverage and development period.

Intended to introduce a sample of large-scale systems development, this paper gives a brief description of what application functions the 'able' system is equipped with, and what systems functions are provided to ensure stable systems operation and to process mass-volume transactions. Also reported here is the progress of the long-term sytems development in addition to some points related to the early detection of problems occurring one after another and the management by which to solve such problems.

1. はじめに

全日本空輸株式会社(以下ANA)の新システム「able」の国内旅客系システム，および新ネットワーク・システム開発を受注し，プロジェクトが発足したのが昭和61年5月であった．当時は，ANA初の国際線サービスである61年3月のグァム線就航を始めとする国際線サービス・システムの開発中の困難な状況下にあり，さらに新しいアーキテクチャを採用した新規システム提案の受注であり，波瀾含みのスタートであった．

内工要員・外工要員とも計画通り充当できず，プロジェクトの体制が整ったのは61年11月であった．その後，仕様確定の遅れ，確定仕様の変更，機能追加要求による作業量増，厳しい作業スケジュール，要員の過残業等，幾多の問題が立ちはだかった．

しかしプロジェクト要員，および関連各位の努力により一つ一つ問題を克服し，国内旅客系システム・フェーズIとネットワーク・システム・フェーズIを昭和63年5月23日にカットオーバした．

引き続き開発作業を推進したableシステム・フェーズII/XTPA(拡張トランザクション処理アーキテクチャ：Extended Transaction Processing Architecture)のカットオーバも平成元年6月1日に達成するに至った．新ネットワーク・フェーズII開発作業を残すものの，今回のカットオーバをもって3年間に及ぶ大規模システム開発の区切りをつけることができた．

この3年間にわたる開発について，ANA新システムの紹介とその特徴，システム開発経緯の概略，およびこの開発におけるプロジェクト・マネジメントのポイントを記述し，大規模システム開発の事例紹介としたい．

2. ANA新システム概要

2.1 ANAシステム構成

ANAでは，多様化する旅客ニーズへの対応や利便性の高い旅客サービスの提供，そして航空会社の使命である安全性・定時性の確保のため，情報系システムが構成されている．

今回の新システムANA総合旅客システム「able」は，ANAの国内線での豊富な実績を基に，さらなる発展を実現するため，「販売力強化による確固たる経営基盤の確保」を目的として開発されたシステムであり，国内旅客系システムと国際旅客系システム，および飛行のオペレーションを司る空港系システムを加えた三つのホストコンピュータと，これらホストコンピュータとを結合し，全世界のネットワークをコントロールするフロントエンド・プロセッサ(FEPシステム)から構成される．これらのシステムの他に，国際貨物システム(USAS＊CGO：Unisys Standard Airline System for Cargo)と整備・管理系システムがある(図1)．

本稿ではこれらのシステムのうち，今回当社が開発を担当したable国内旅客系システムとFEPシステムの開発について記述する．

2.2 able国内旅客システムの業務機能

本システムは，座席予約→発券→搭乗→精算→実績までの流れを管理することができるトータル・システムである．また，航空券と搭乗券が一体となった磁気ストライプ付きの航空券(ATB券：Automated Ticket & Boarding Pass)を全面的に採用した世界で初めてのシステムである．

本システムの主たる新規業務機能は次の通りである．

2.2.1 予約・発券業務

1) 全席事前座席指定(ASR：Advanced Seat Reservation)……航空券購入時に，全席「前方」「禁煙」「窓側」等，希望座席を指定できる．
2) 搭乗券一体型航空券(ATB券)……これまでの航空券と同サイズで，磁気ストライプの情報によりチェックインの容易性と事後処理の効率化を図っている．
3) 自動発券対象の拡大……手作業で発行されていた団体航空券等，ほぼ全種類の航空券を自動発券機によりATB券として発行する．

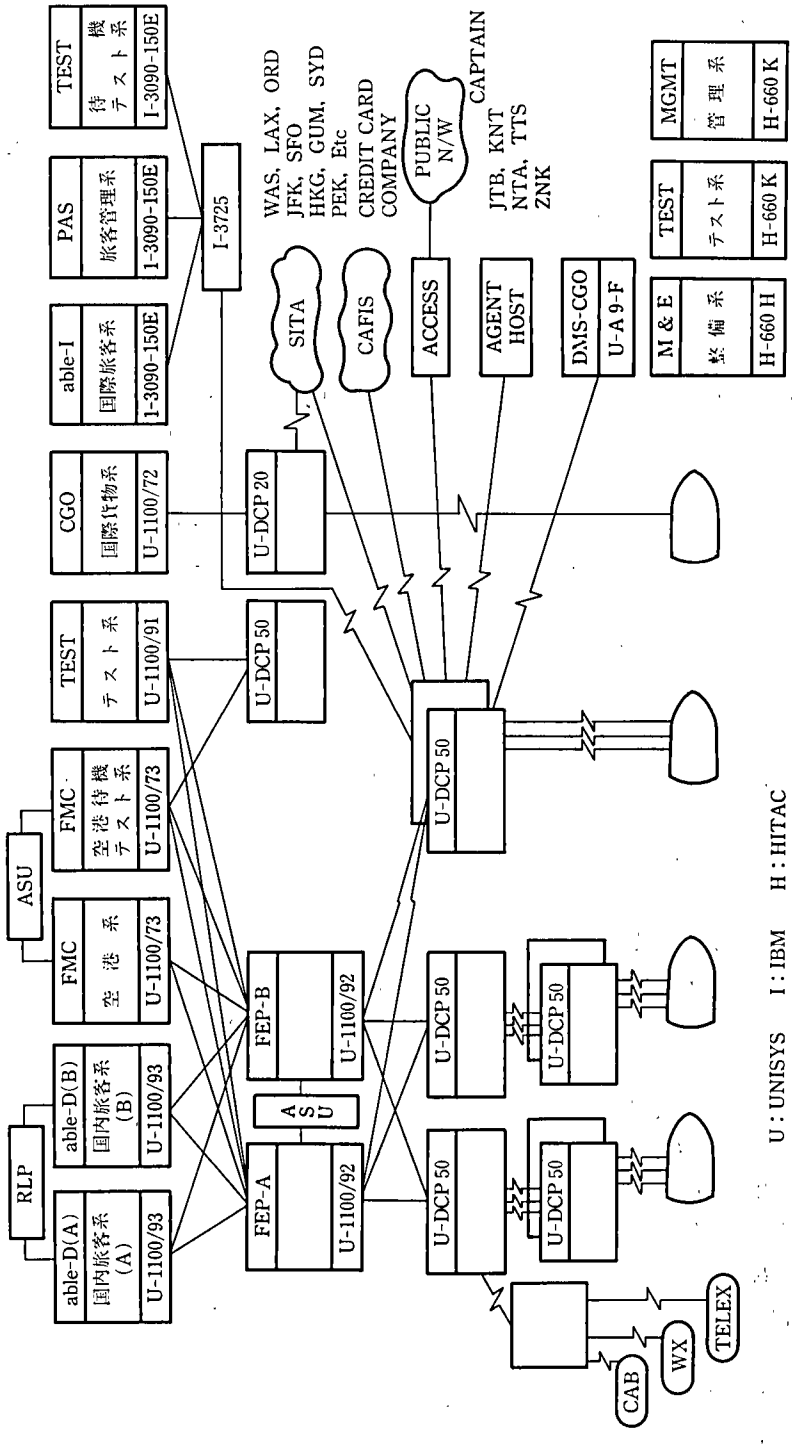


図1　ANA システム構成図

Fig. 1　ANA system configuration

4) 簡易発券機(able-Q 端末)……公衆回線を介してホストコンピュータと接続し，ATB 券発行等の機能を備えたコンパクトな簡易端末の利用により販売網の拡大を図っている．

5) 自動券売機(ATV：Automated Ticket Vender)……顧客自ら簡単なタッチ操作により，現金あるいはクレジットカードで航空券を購入し，チェックインが同時に行える ATV を各地の空港に設置し利便性を図っている．

6) その他の機能として，漢字による一般情報の表示，精算業務システムの強化等がある．

2.2.2 搭乗手続(チェックイン)業務

1) ANYWHERE チェックイン……行き先にかかわらず，チェックインがどのカウンタでも行えるため，カウンタの混雑解消や短時間でのチェックインを実現している．

2) ONE POINT チェックイン……航空券購入からチェックイン，手荷物の預かりまでを一つのカウンタで行える機能で，カウンタの混雑解消や短時間でのチェックインを可能にしている．

3) セルフチェックイン(SCM：Self Check In Machine)……空港カウンタ横の自動チェックイン機で，顧客自身が ATB 券を挿入し，スムーズにチェックインができる．

4) 発券，即チェックイン……出発当日に空港で航空券を購入した場合，発券カウンタで航空券の購入と同時にチェックインができる．

5) その他の機能

① オンライン大型案内表示盤(IED：Industory Electronic Display)のサポート

② 乗り継ぎ便のチェックインを1回で完了させるためのスルーチェックイン機能

③ 手荷物タグ(Baggage Tag)のオンライン自動発行

④ センタ・コンピュータのトラブル時，主要空港に設置しているミニコンでの簡易チェックインを可能とするためのバックアップ・システム

⑤ 駅，ホテルでのチェックインができるシティ・チェックイン機能

⑥ 自動改札機(AGC：Automatic Gate Counter)の採用，等

2.2.3 搭乗載管理(LCS：Load Control System)業務

飛行機の安全運行のため，旅客や手荷物・貨物・燃料等の重量を飛行機の総重量をもとに最適な重量とバランスを計算するシステムである．

2.3 システム機能

365 日 24 時間の無停止オンライン・サービス，安全運転のための相互バックアップ，センタオペレーションの効率化，あるいは大量トランザクション処理実現等のため，本システムは各種のシステム機能を有している．

主な機能は次の通りである．

1) 安全運転のための相互バックアップと大量トランザクション処理実現のため，国内旅客系システムにおいては世界初の XTPA での2ホストシステム，ネット

ワーク関連ではマルチ FEP システム，マルチ DCP 50 システムと単なる待機システムではなく，資源の有効利用を目的とした二重化構成をとっている．

これらの機能は，トランザクションが大量に発生した場合でも，各システムがバランス良く処理できるように，国内旅客系ホストシステムと FEP システム間，DCP 50 と FEP システム間でのトランザクション流量制御(フロー・コントロール)や，優先トランザクションのレスポンスが確保できるような工夫がしてある．

2) センタオペレーションの効率化，簡素化のために，複数システムのコンソール・オペレーションが一箇所でできることを可能にした集中制御システム(CONSOLE 1100)，各システムが正常に稼働しているかどうかを監視する集中監視システム(MONITOR 1100)を用意している．また，万一障害が発生してもできる限り自動回復を行うように相互監視システム(SAFE 1100)を採用している．

3) ネットワーク・システム関連機能としては，大規模な able 端末接続を始めとして，SNA ゲートウェイによる IBM ホストシステム(国際旅客系システム)との接続，SITA(国際航空通信共同体：Société Internationale de Télécommunications Aéronautiques)接続による海外端末サポート，代理店やクレジット会社等の対外システムとの接続等バラエティに富んだネットワークをサポートしている．これらの大規模ネットワークの一元管理(構成管理，障害管理，総計情報管理，構成機器管理)のために TNAS(Total Network Administration System)がある．また，able システムを構成する各ホスト間の情報交換ができるような通信機能も有している．

4) テストや端末トレーニングが効率良く，そして実環境に近い形態でできるように，国内旅客系システム内に当該データベースを共存させたり(テストグループ)，DSF(Development Support Facility)を中心とした各種開発，テスト支援ツールを整備している．

2.4 XTPA システム

ANA 新システムは，大量トランザクション処理とノンストップ・システム実現のため，ユニシス独自のシステム・アーキテクチャ「XTPA」を採用した世界で初めてのシステムである(図 2)．

XTPA は OS 1100* を拡張した「OS 1100/XTC**」と「RLP(レコードロック・プロセッサ)」により具現化されている．

また，ハードウェア，アプリケーション，データベース等，一部の制限を除き既存のリソースはそのまま XTPA のもとで使用できる．

2.4.1 XTPA の機能概要

XTPA の主な機能は次の通りである．

1) 複数ホストシステム間で同一データベースの共有が可能である．

2) RLP でレコード単位のロック・コントロールが可能である．

3) RLP による高速ロック・ハンドリングを実行する．

4) OS 1100 の MHTIP(Multi Host Tip)機能で，TIP(Transaction Interface

* OS 1100：UNISYS 1100/2200 シリーズのオペレーティング・システム

** XTC：eXtended Transaction Capacity

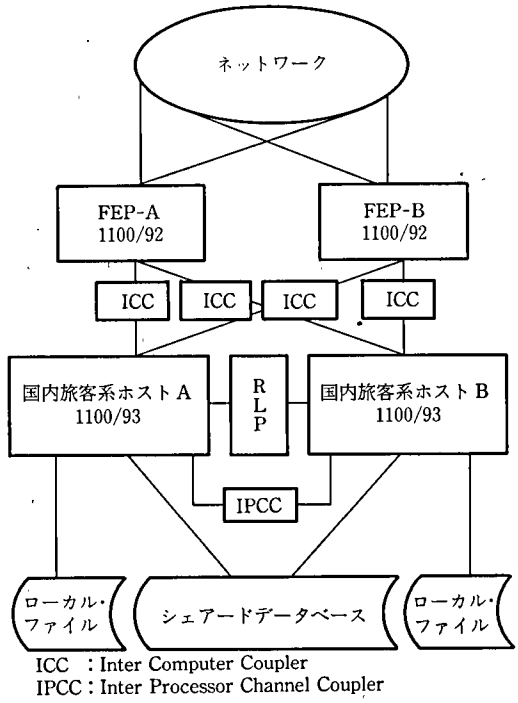


図 2 ANA XTPA システム概略図

Fig. 2 The outline of the XTPA system

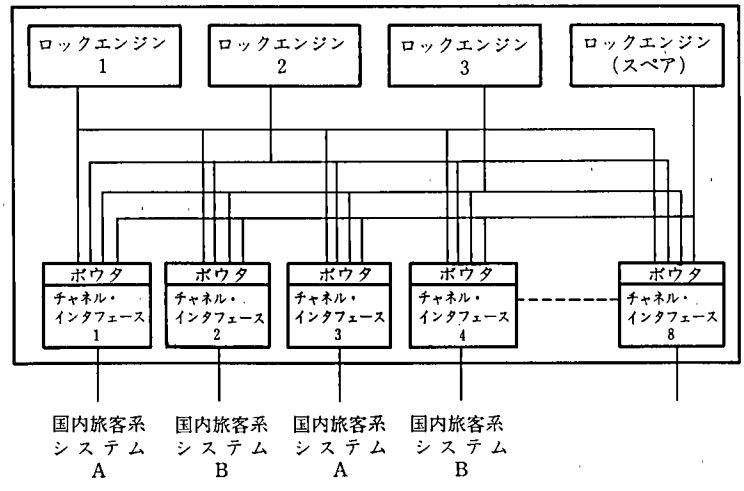


図3 RLP 概略図

Fig. 3 The outline of Record Lock Processor

Package)データ構造の変更がホスト間伝達できる.

2.4.2 大量トランザクション処理の実現

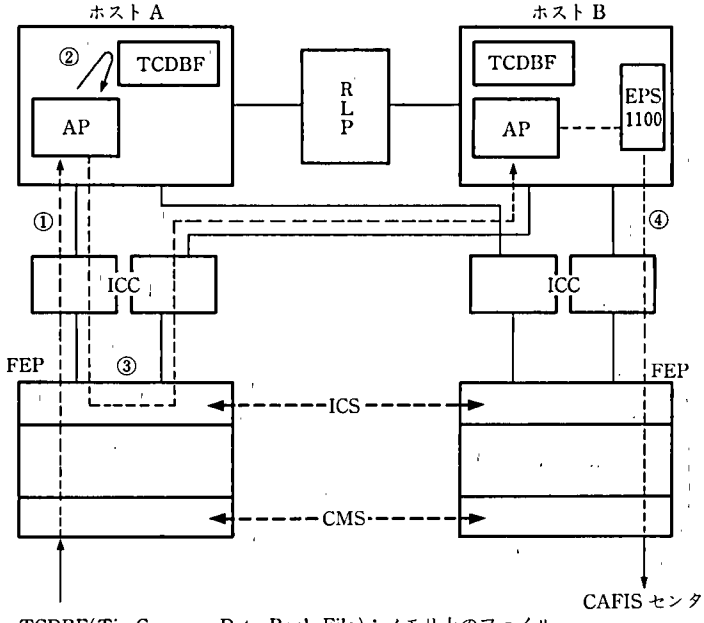
前述の機能により,XTPA 内の複数ホストシステムで同じアプリケーションの実行ができ,単一ホストシステムでさばききれなかった大量トランザクション処理に対応することができる.XTPA のもとで UNISYS 1100/93×2 セットの able 国内旅客系システムは,秒間数百件のトランザクション処理を実現している.

当初,懸念された当該システム採用によるベーシック・ソフトウェアのオーバヘッドも良好な結果を得た.また,データベースをシェアしながら二つのホストシステム間で異なったアプリケーションも実行できるため,夜間に実行するナイトリー・ファイル・メンテナンス処理(夜間バッチ処理)等をラン(ジョブ)ごとに別ホスト・システムで実行させ負荷分散により処理時間の短縮を図ることができる.

なお,ANA 新システムで大量トランザクション処理を実現している XTPA とは別のもう一つのポイントは,ハイスピードの大型 SAS(Semiconducter Auxilialy Storage)を使用し,入出力処理の効率を上げたことである.

2.4.3 ノンストップ・システムの実現

複数のホストシステムが同一のアプリケーションを実行し,同一のデータベースを更新する.そして,データベースの整合性を保つため同一のレコードを複数のシステ



TCDBF(Tip Common Data Bank File):メモリ上のファイル

① 入力トランザクションを処理したところ CAFIS センタへ問い合わせ発生.
② CAFIS センタとのインタフェースをとるソフトウェア EPS 1100 がどちらの系で稼働しているかチェックし,他系での稼働を確認.
③ ホスト B 系に CAFIS センタへの問い合わせを依頼.
④ CAFIS センタへデータ送付.

図4 片系のみで稼働するアプリケーションの運用

Fig. 4 The flow of application that operated in the only one side

ムが同時にアクセスをしないように，レコード単位で常に RLP が管理する．このため万一，一方のホストシステムで障害が発生しても，正常ホストシステムは何ら影響を受けることなく処理の続行ができる．また，RLP 自体の部分障害が発生してもスペア・モジュールへの自動切り換え，あるいはパスの二重化等，安全対策が確保されている(図 3)．

2.4.4 XTPA システム実現のための考慮点

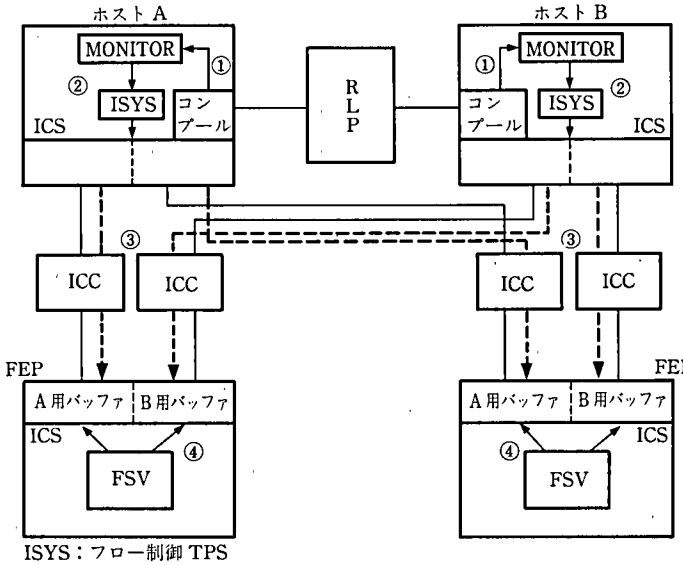
XTPA システムの構築において，いくつかの考慮点が発生した．

その代表的なものは以下の通りである．

1) 片系ホストシステムでのみ稼働するシステムの運用……ソフトウェアの特性により，片系ホストシステムのみで運用しなければならないアプリケーションがあった．その代表的なものとして，クレジット会社(CAFIS センタ)との接続システムがあり，図 4 のようなシステム上の対応を実施した．

2) トランザクションのロード・バランス……何らかの原因により，ホストシステムのトランザクション処理バランスがくずれる場合がある．その状況が発生した場合，ただちに処理バランスを保つことができるようにフロー制御機能を開発した．その機能概要を図 5 に示す．

3) 片系ホストシステムの障害時リカバリの自動化……ノンストップ・システム実

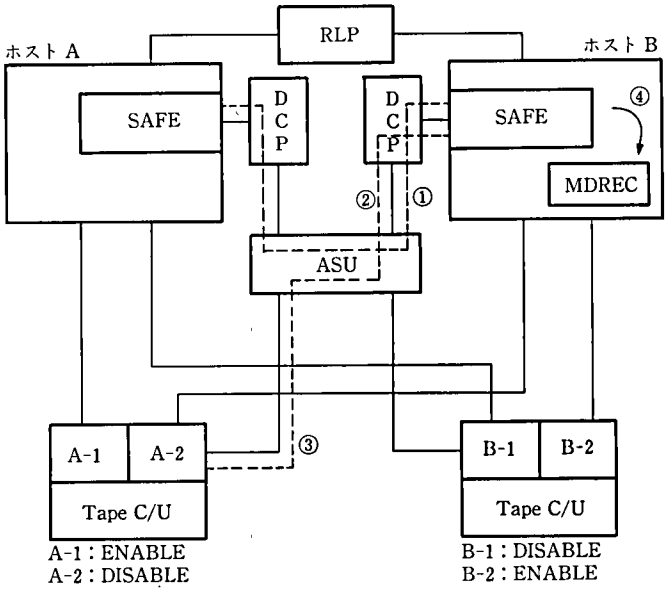


ISYS：フロー制御 TPS
FSV ：FEP サービスモジュール

① 入力コンプール (COMPOOL, Common Data Pool) の使用状況を監視しフロー制御レベルを決定する．
② フロー制御レベルに対応した FEP からの送出件数値を各 FEP へ伝える．
③ 各 FEP ごとに対局ホストに対する送出件数値をテーブルに設定する．
④ 各ホストへの送出件数値に達するまで，各ホストのバッファにトランザクションをセットする．

図5 トランザクションのフロー制御

Fig. 5 Transaction flow-control



ASU：Automatic Switching Unit

（ホスト A 障害の場合の処理）
① SAFE の相互監視により，A 系障害を検知。
② SAFE から ASU に切り換えを指示(ディスクおよびテープ)．
③ ASU はホスト A の Tape C/U に対し，
　A-1：ENABLE → DISABLE
　A-2：DISABLE → ENABLE
　に変更を行う（テープ自動切り換え）．
④ ホスト A 系のリカバリ・ラン（MDREC）を起動し，リカバリを実施．

図 6　片系システム障害時リカバリの自動化

Fig. 6　The outline of automatic recovery

現のための代表的な機能としてこの障害時運用がある．相互監視システム(SAFE 1100)との連動により，当該状況時のリカバリ運用の自動化が図られている．その概略の流れは図 6 の通りである．

3.　システム開発経緯

3.1　システム開発スケジュール

昭和 61 年 5 月，新システム開発に着手した．開発期間はフェーズⅠシステムが 24 か月であった．さらに，その後 1 年後にフェーズⅡシステムの稼働が予定されていた．作業量見積りで 6,000 人月以上のシステム規模・範囲から見て，大開発にもかかわらず厳しい期間が設定された．開発要員の確保も簡単にはいかず，最終的に開発体制が整ったのは半年遅れの 11 月であった．また，稼働開始の 3～4 か月前からトレーニングが始まるため，実際の開発期間はさらに短かかった．

一方，フェーズⅡシステムもフェーズⅠシステム開発のピークであった昭和 62 年 4 月に，開発のための組織編成を行い作業に着手した．仕様確定の遅れ，確定仕様の変更，機能追加要求による作業量の増大，厳しい作業スケジュールとその一部遅延，要員の過残業，協力会社メンバの確保・管理，作業スペースの確保，新センタへの引越

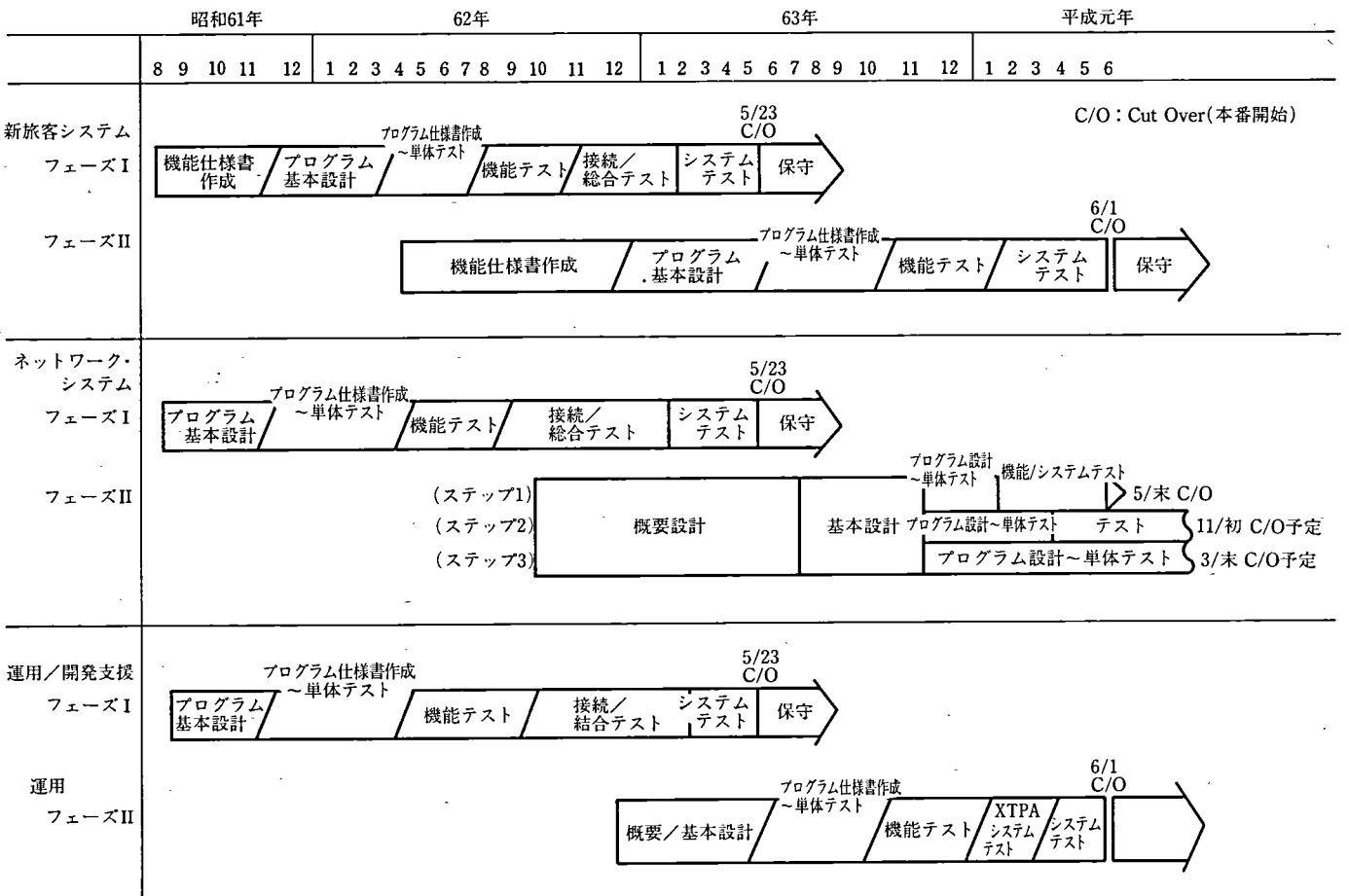


図7　ableシステム開発スケジュール

Fig. 7　Outline of the schedule that able system development

し，さらには，稼働システムの保守，運用サポート，解決しなければならない問題は山積みであった．しかしながら，これら幾多の問題を一つ一つ解決しながら図7のスケジュールで機能削減することなく予定通り開発作業を終了した．

3.2 プロジェクト要員数

プロジェクト要員数の推移は図8の通りである．ピークはフェーズⅠシステム開発の詳細設計，プログラミング，単体テスト期間で，フェーズⅡシステム開発作業を着手した時期でもあった．この時のプロジェクト要員は約350名であった．機能テスト段階とシステムテスト段階で要員の減少があり，その後は200名強の要員でフェーズⅡシステムのカットオーバまで推移した．なお，この間の協力会社は50社にも及んだ．

3.3 アプリケーション・システム開発規模

able システムのアプリケーション部分の開発規模，および基本設計からシステム・テストまでの作業工数を表1に示す．

この結果を簡単に説明すると次の通りである．

1) 開発ステップ数の見積りと実績の差異……原因は，機能仕様の追加・変更だけ

表1 アプリケーション・システム開発規模
Table 1 A scale of the application systems

	フェーズⅠ（昭61.5～昭63.5）				フェーズⅡ（昭62.4～平元.5）			
	ステップ数		工数（人月）		ステップ数		工数（人月）	
	見積り	実績	見積り	実績	見積り	実績	見積り	実績
システム移行／改善（SYS, RES, MSG 他）	54,000	135,000	1,800	2,500	—	—	1,800	2,000
販売系システム								
予約関連（SYS, RES 他）	69,000	90,000			36,000	45,000		
発券関連（TKT）	14,000	19,000			70,000	72,000		
運送系システム（CKI）	45,000	72,000			87,000	122,000		
搭乗戒管理システム（LCS）	86,000	100,000			—	—		
合計	268,000	416,000	1,800	2,500	193,000	239,000	1,800	2,000

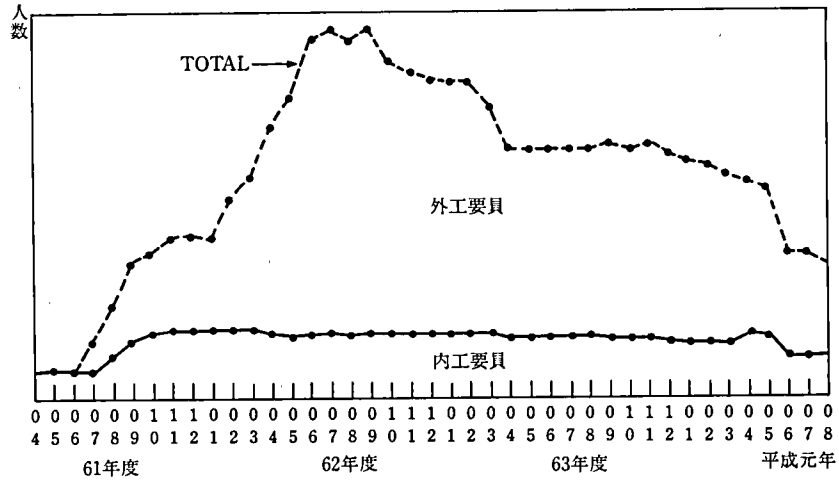


図8 年度別プロジェクト要員数
Fig. 8 The number of the project members by years

でなく，仕様の曖昧さによるものも一部あった．とくにフェーズⅠシステム移行・改善部分は後者の原因によると考える．またフェーズⅠよりフェーズⅡの差が少ないのは，フェーズⅠの経験と仕様がある程度固まった後の見積りであったためと思われる．

2) 生産性(ステップ数/工数)……フェーズⅠとフェーズⅡの生産性を比べると見積り，実績ともフェーズⅠの方がやや高い．これはフェーズⅠのLCSが新規開発であり，その他は改修開発のためである．

4. プロジェクトのマネジメント

今回の新システム開発で直面した問題点，あるいは苦慮した点を以下に列挙する．これらは客先システムの開発業務を受託した場合，多かれ少なかれ共通するところと思われる．

- 開発システムの大規模・複雑化
- 短期間のシステム開発
- システム要件の変化に対応した開発
- 複数システム系の関連性を留意した開発
- 他メーカの開発システムとの接続
- 社会性を鑑みた品質確保
- 将来の技術水準に対応したシステム構築
- 開発要員の確保
- 開発要員のスキルのアンバランス
- 客先，あるいはプロジェクト要員間の共通認識不足

これらの問題点を一つ一つ解決してシステム開発を成就したわけであるが，このためのマネジメントの基本方針として「目に見えるシステム開発作業の推進(Visible Development)」を掲げた．

4.1 プロジェクト・マネジメントの方針

客先システムの開発をする場合，プロジェクト要員間はもとより，客先と共通認識を持って作業を推進する必要がある．しかしながら，利害関係あるいは立場上の違いから，必ずしも円滑なシステム開発作業ができないことがある．とくに，大規模システム開発作業においては一般に，作業範囲も広く，開発期間も長く，また多くの要員が参加するため顕著である．

この問題の解決策として，過去のシステム開発の経験を生かした目に見えるシステム開発を推進した．これは開発作業の進捗状況を正確に把握し，作業上の問題点の早期発見と，その本質・適確な解決策を明確に示すことである．そしてその評価結果を常にドキュメント化することにより，プロジェクト内および客先と共通認識を持ち，各種オペレーションを円滑にするわけである．また，目に見えるシステム開発の推進は，プロジェクト要員の共通の達成感，マインドの高揚とともに，システムの品質向上にも寄与したと考える．

この考え方を踏まえたマネジメントの主な方針は次の通りである．なお，具体的に実施した各種管理は次節に述べる．

1) 作業計画と実績管理の徹底……システム開発において作業の進捗管理は基本である．この作業進捗状況の把握と評価，問題点の把握と対処等を効果的に実施するため3種のスケジュールを策定し管理した．一つは，開発作業の開始時に作成するマスタ・スケジュールで完了までの基本作業項目を洗い出し，その作業予定を設定する．

その他にマスタ・スケジュールに準拠し，作業項目を具体化して作業担当予定者を含め3か月ごとに作成するクォータリ・スケジュールと毎月作成するマンスリ・スケジュールがある．マンスリ・スケジュールは作業項目をさらに分解し，作業担当者と作業期間が設定される．クォータリ・スケジュールとマンスリ・スケジュールの策定はローリング方式で行ったためマンスリ・スケジュール設定には数回のチェックが入り，作業項目の抜け防止，個々の作業スケジュールの整合性保持等に役立った．これらのスケジュール評価，実績管理のため，週一回のプロジェクト・レビュの継続的実施を徹底した．

2) 計数管理に耐える開発作業の実施……前述の作業進捗状況の計数表示，障害発生件数とその解決件数，作業工数の予定と実績，開発コード量等，マネジメントに必要なデータをできる限り，計数表示，グラフ化を行い実績把握が容易に，かつ正確に行えるものにした．

3) 品質向上と生産性向上……この問題の解決策の一つとして，ドキュメント化の徹底を行った．システム開発において，機能仕様書・設計書・テスト計画書等のドキュメント作成は当然であるが，その他ドキュメント化可能なものはできる限り対応し工夫した．たとえばテストを実施するに当たって，そのテストケースを記述し，その結果を評価・報告できる'TCF'(Test Case File)や，仕様の曖昧さを回避し確認するために使用される'TQ'(Technical Question)等である．

また，デザインレビュ，コードレビュを徹底すると同時にウォークスルーによる品質向上も実施した．その他ソフトウェア管理や，開発支援ツール，テスト環境の整備改善等をマネジメントの方策に掲げた．

4) プロジェクト要員の共通認識の確立……前述のスケジュール管理による作業進捗，計数による状況認識だけでなく，同時に技術情報を民主化すること等により，ピーク時350名を超えたプロジェクト要員が共通認織を持つことができるように対応した．

4.2 開発管理

4.2.1 プロジェクト管理

円滑なシステム開発作業の推進のため，作業進捗状況の把握，問題の本質と解決すべきポイントの明確化等，プロジェクト要員はもちろん，客先や社内関連部所との共通認識を図るためのマネジメントである．

以下に具体的な管理システムとそのツールおよび運用の概略を述べる．なお，プロジェクト管理の概略を図9に示す．

1) スケジュール管理……開発作業分野ごとの作業進捗状況の評価・問題点の把握および対処等のために，MAPPER システムを構築した．MAPPER システムには，スケジュールの登録・変更・削除・進捗率の更新機能を持たせた．

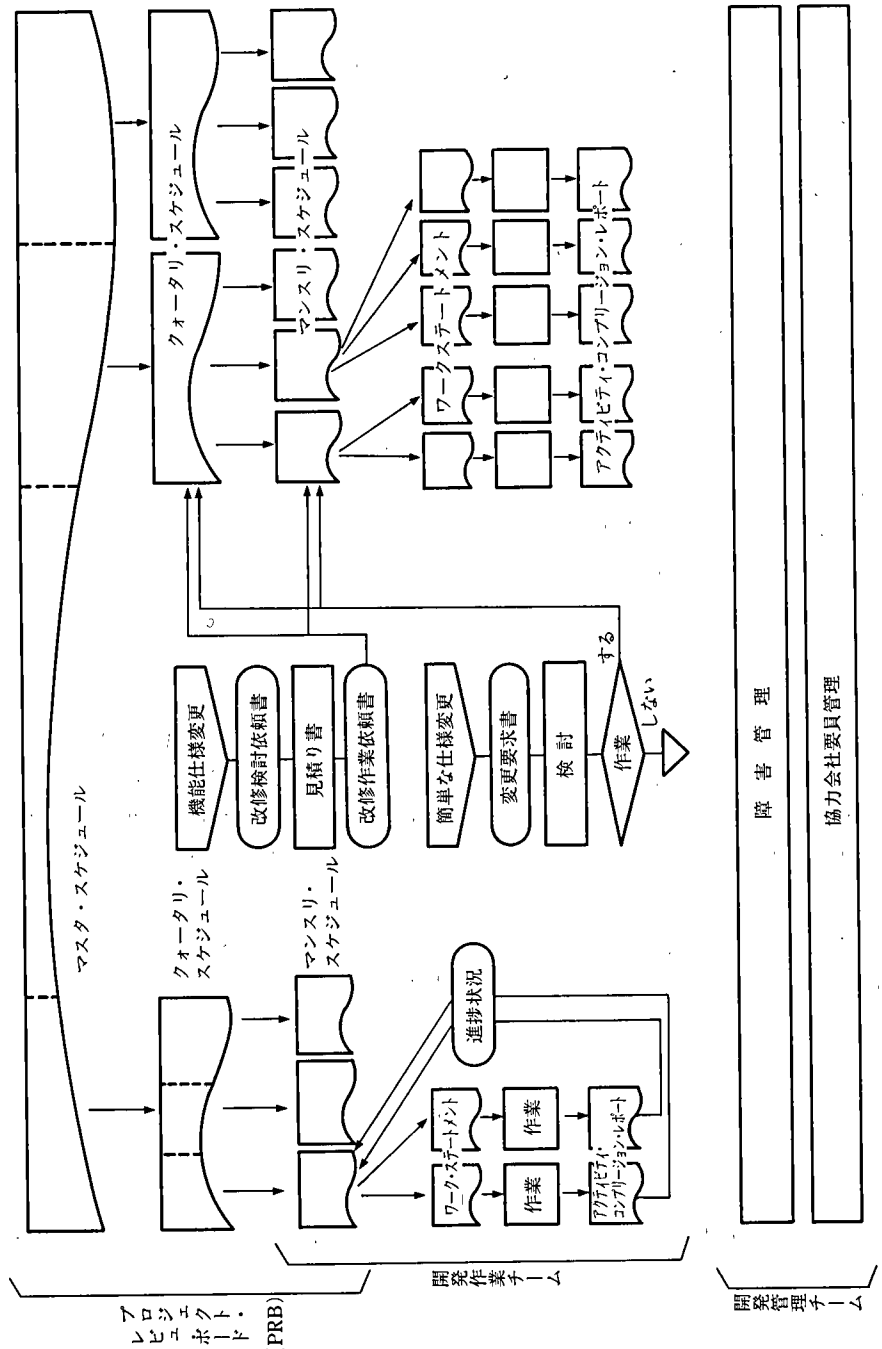


図9 プロジェクト管理概略図

Fig. 9 The outline of the project management

管理するスケジュールの種類は，マスタ・スケジュール，クォータリ・スケジュール，マンスリ・スケジュールである．これらのスケジュールを基に週1回，部長・課長で構成したプロジェクト・レビュ・ボード(PRB：Project Review Board)で継続的な審議の実施を徹底した．前節で述べた通り，クォータリ・スケジュールとマンスリ・スケジュールの策定をローリング方式で行うことにより，その効果を高めている．

2) アクティビティ管理……マンスリ・スケジュールにて策定した作業項目単位の管理を行うため，スケジュール管理と連動して次の2種類のレポートが出力される．

① ワーク・ステートメント：作業指示書と報告書を兼ねるレポートで作業進捗状況把握の精度を上げる．

② アクティビティ・コンプリージョン・レポート：作業実績を記入した作業完了報告書で，以後の作業スケジュール策定のための参考データともなる．

3) 変更管理……開発途上で発生する変更要求に関して，開発計画に対する影響を極小化するとともに，作業の混乱を少なくするために客先間あるいは部内の開発作業チーム間で組織的に実施するものである．これらのツールとして，変更管理台帳・変更要求書(RFC：Request For Change)・変更作業計画書・変更作業完了報告書がある．また，変更作業の実施可否の決定機関として，客先およびプロジェクト内にRFCボードを設定し運営した．

4) 障害管理……稼働システムおよび開発中のシステムで発生した障害の追求状況，追求結果を管理・分析するためのもので，規程を設定し運用した．なお，統計資料の作成が容易となるよう当該システムもMAPPER化している．

5) 改修管理……稼働システム，あるいは開発システムの機能拡張，仕様変更等で発生する改修業務を実施するに当たって定めた管理である．ここでの改修とは客先要求に基づく受託業務であり，会社対会社の契約事務と直接的に連動する．運用(手続)は次の通りである．

- 改修検討依頼書：客先より提出される見積り依頼
- 見積り書　　：改修検討依頼書に対する客先への回答書
- 改修作業依頼書：客先からの正式な作業依頼書
- 改修完了報告書：作業を完了した時に客先へ提出

6) 会議の運営と管理……プロジェクト管理の社内会議として，以下の3形態を設定し運営した．

① プロジェクト・レビュ・ボード(PRB)：プロジェクト内の部長・課長より構成されるボードで毎週開催される．ここでは，各課長よりの報告書で作業進捗状況の把握，問題点の対処案，作業チーム間の調整等，プロジェクト全体の状況を確認する．また，変更管理のためのRFCボードも兼ねている．

② 開発作業チーム会議：作業項目に対応した作業チーム単位の担当課長がPRBの前に開催する会議である．作業担当者の作業進捗状況をレビュし，チーム全体の調整を行う．

③ テクニカル・レビュ・ボード(TRB)：開発途上で発生する技術的な問題を

検討し，解決案を策定する会議形態で原則的に週一回開催した．なお解決のために作業が発生する場合，その作業担当，スケジュール等の決定はPRBにて行われる．

これらの社内会議とは別に，客先と合同で行う3形態の会議を次に示す．

④ 週次進捗報告会議：客先と当社の対応する作業チームごとに，週一回開催され，それぞれの作業進捗状況の報告，問題点の原因と対処法，翌週の作業予定等を確認・調整する．

⑤ 月次進捗報告会議：毎月初めに開催する客先プロジェクトとの合同会議である．双方からの提出資料により，作業進捗状況，問題点とその原因と対処法，および翌月度作業予定等を確認・調整する．

⑥ 合同プロジェクト会議(メーカ会議)：作業フェーズの大きな節目ごとに実施される会議で，当社からは，関連本部長・部長，客先からは部長・課長等の上位マネジメントが出席する最高決定機関である．マスタ・スケジュール・レベルでの作業進捗状況，次工程への作業推進等を確認・決定する．なお，該会議の前に調整会議を実施し，マスタ・スケジュールの変更等の問題がある場合には，双方協議の上，合同プロジェクト会議への答申案を策定する．

7) レポート(アナウンスメント)……プロジェクト内の共通認識徹底の方策の一つとして，テストステータス，技術情報等の迅速な報告・伝達のために，次のような標準用紙を定め運用した．

① Daily Development Status Report：機能テストやシステム・テスト等，各開発チームが相互に絡み合うテストにおいて，その結果をタイムリに伝達する場合に使用する．

② System Memorandum：共通プログラムの効率的な使用方法，あるいは制限事項が判明したとき等，プロジェクト全要員にその情報を伝える場合に使用する．

4.2.2 ソフトウェア開発管理

作業の成果物，すなわちソフトウェアの品質・機能・生産性向上のための管理がソフトウェア開発管理である．作業進捗管理のためのプロジェクト管理同様，システムの開発管理において重要なポイントであることは言うまでもない．これらの目的のために標準化し，実施した管理を述べる．

1) 開発基準管理……ソフトウェア開発の生産性・品質向上のため，次の基準書を作成し，標準化を行い開発要員のガイドとした．

・システム設計基準書(プログラム設計者の指針となる基準書)

・プログラミング基準書(プログラマのためにプログラム作成規則やコーディング方法を定めた基準書)

2) 作業標準管理……ソフトウェアの品質向上を主眼とし，開発要員の共通理解・認識を深めるための管理で，下記の各種ドキュメント作成要領・レビュ要領・定型用紙等を設定し，それぞれの運用を徹底した．

・開発文書管理：プログラム基本設計書，プログラム仕様書，テスト計画書/報告書等の作成指針を定めたものである．

・レビュ要領書：デザインレビュ，コードレビュのための指針を定めたものである．
・TQ：仕様の曖昧さの回避のための用紙である．
・CCF(Change Control Form)：プログラム・コード管理のための用紙である．
・TCF：テスト実施に当たってテストケースを作成するための標準用紙である．

3) ライブラリ管理……ライブラリ管理ツールとして，USAS * AIA(Application Installation Aid)の SCCS(Source Code Control System)を使用し，開発システムのプログラムレベル，プログラム・コード・ヒストリ等の管理を行った．

4) ソフトウェア・リリーズ管理……ソフトウェアのリリーズについてルール化をするとともに，リリーズに先立ち参加要員の共通認識のため，Software Release Announcement および Software Release Memo のレポートを配布した．

5. おわりに

平成元年6月1日，システム切り替え作業の「長い夜」が明け，able システムが誕生して4か月が経過した．この間，若干のトラブルはあったが，安定稼働をしている．アプリケーション機能，XTPA の安定性，パフォーマンス等のシステム設計目標(2.3節)も達成できた．

この3年間を振り返ると，非常に苦しい作業の連続であり，とくにフェーズⅠシステム開発においての，仕様変更，曖昧さによる作業スケジュールの遅延，テストツールの性能不備，パフォーマンス不足等，問題は山積みであり，その対応で深夜に及ぶ作業の連続であった．これらの経験が生きて，フェーズⅡシステム開発作業は厳しい作業であったもののフェーズⅠに比べると比較的順調に推移したと言える．

長期にわたる開発作業における幾多の問題，あるいは厳しいプロジェクト・マネジメントの中，一つの大きな仕事が成就したのは，プロジェクト要員の粘り強い継続的な努力と関連各位の強力なサポートの結果である．また，この大規模システムのカットオーバを経験したことにより，プロジェクト要員一人一人が各々の立場で果たすべき役割と発揮すべき技術および開発マナーを身に付けた．さらに，何ものにも替えがたい達成感をエネルギに，新たなるシステム開発に取り組めるものと確信している．

なお，プロジェクト・マネジメントの記述については，当該システム開発の責任者であった，本池　洵氏(現在，システム第一本部システム八部部長)の社内資料を参考にした．

執筆者紹介 中　島　　崇(Takashi Nakajima)

昭和21年生．44年青山学院大学理工学部経営工学科卒業．同年日本ユニシス(株)入社．社会公共関連(北海道電力，日本旅行，労働省，全日空)のシステム開発，SEサービスに従事．現在，システム第一本部システム七部(エアラインシステム)に所属．

ALMシステム（TOPPS/ALM）の開発

Development of the TOPPS/ALM System

宮田敏光

要約 ALM（Asset/Liability Management：資産・負債総合管理）は，金融自由化を乗り切るために米国の金融機関で取り入れられた経営管理手法（リスク管理）であり，金融自由化が進展している日本においてもALMの必要性が高まってきている．

ALMは，経営計画，予算編成・収益管理，原価管理とともに金融機関経営の中核となるものであり，当社では，これらの業務を総合的に支援する「総合利益計画管理システム」（TOPPS：Total Profit Planning System）の開発・提供を行っている．

ALMシステムは，「総合利益計画管理システム」を構成するシステム群の一つである．このALMシステムを，足利銀行との共同開発により約1年半の期間で開発し，パッケージソフトウェアとして販売を行っている．今回のALMシステム構築に当たっては，CITIBANKからALMに関するさまざまなノウハウを受けている．

本稿では，このALMシステムの概要とシステム開発をどのように行ったか，どのような点に留意したかを紹介する．

Abstract ALM (Asset/Liability Management) is one of the business administration methods (risk management) adopted by financial institutions in the United States, who want to get over the waves of financial deregulation, and the necessity for ALM has also been drawing more and more attention in Japan where there is financial deregulation now in progress.

ALM, along with business planning, budgeting, B/S and P/L planning and cost accounting, plays a central role in the management of financial houses. To support these applications, the total profit planning system dubbed TOPPS has been developed by Nihon Unisys, Ltd. for release to customers.

The joint development by both the Ashikaga Bank and Nihon Unisys of the ALM system, which is one of the TOPPS subsystems, took about one year and a half for current sale as a packaged software product. CITIBANK. N. A. has been very helpful in providing a wide range of ALM-related know-how and expertise for the construction of the ALM system now available in Japan.

This paper describes a rough picture of the ALM system, how it has been developed and where attention has been paid.

1. はじめに

「総合利益計画管理システム」（TOPPS）の一環として，ALMを強力に支援するためのシステムを，CITIBANKのノウハウを受け，足利銀行と共同開発した．このALMシステムは，現在数多くの金融機関に導入されている．

本稿では，今回開発したALMシステムの概要と，このシステム開発をどのように行ってきたかを紹介する．なお当初のシステムは，COBOL換算で約28万ステップの規模である．今やこの規模のシステムは大きい方ではなくなってきているが，今後の大規模システム開発の参考になれば幸いである．

2. ALM システムとは何か

2.1 ALM システムの開発の背景

70年代の米国は金融の自由化が進み，急激かつ大きな金利上昇・下落の波に見舞われた．金融の自由化は，銀行に金利の自由化をもたらし，また証券会社等，他業態との激しい競争を余儀なくされるようになった．これに伴い，銀行は流動性リスクと金利リスクという今までになかったリスクにさらされるようになり，従来の経営管理ではこれに対応できなくなっていった．このような状況の中でリスクマネジメントとしての ALM（Asset/Liability Management：資産・負債総合管理）が多くの銀行に取り入れられ，現在に至っている．

日本においても，金融の自由化の進展に伴い，ALM を導入する必要性が高まってきている．当初は有価証券を対象とした部分的な ALM であったが，現在は全資産・負債を対象とする全面的な ALM へと展開されてきている．

ALM を実施するに当たっては，現在の資産・負債の状況がどうなっているかを科目・残存期間・基準金利等，さまざまな切口で詳細に把握しておくことが重要となる．これを行うためには，現状を正確に反映した精緻な ALM データベースの構築と分析システム，すなわち ALM サポートシステムが必要となる．都市銀行や信託銀行では全体 ALM のサポートシステムが稼働し始めているが，多くの金融機関ではサポートシステムの開発・導入はこれからという状況である．ALM システムの導入ニーズは非常に高い．

ALM は今までにない新しい分野であり，どのようなアプリケーションを作り，それをどう活用していくか，というノウハウが重要となる．当社は，CITIBANK と提携し実践的な ALM のノウハウを得て，足利銀行と共同で ALM システム（パッケージシステム）を開発し販売を行っている．

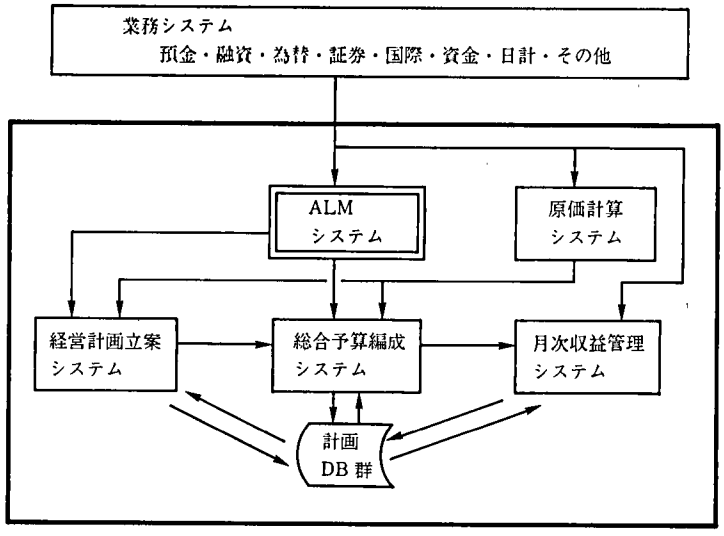


図 1 総合利益計画管理システム（TOPPS）

Fig. 1 Total profit planning system

2.2 システム概要

経営計画，予算編成・収益管理，ALM，そして原価管理は，金融機関における経営情報システム構築上の最重要課題となっている．当社では，これらを「総合利益計画管理システム：TOPPS」として体系付け，システムの開発・提供を行っている．ALM システムは，この「総合利益計画管理システム」を構成する1サブシステムとして位置付けられる(図1)．

ALM システムは，以下の三つのサブシステムに分けられる（図2）．

1) ALM データベース作成システム……個別取引明細データから ALM 基本データベースを作成・更新する．
2) ALM 計数把握システム……ALM 基本データベースに基づき，現状の資産・負債の構造がどうなっているか分析する．
3) 意思決定支援システム……資金シナリオ，金利シナリオをもとに，将来の収益予想や資金調達計画を作成，政策立案を支援する．

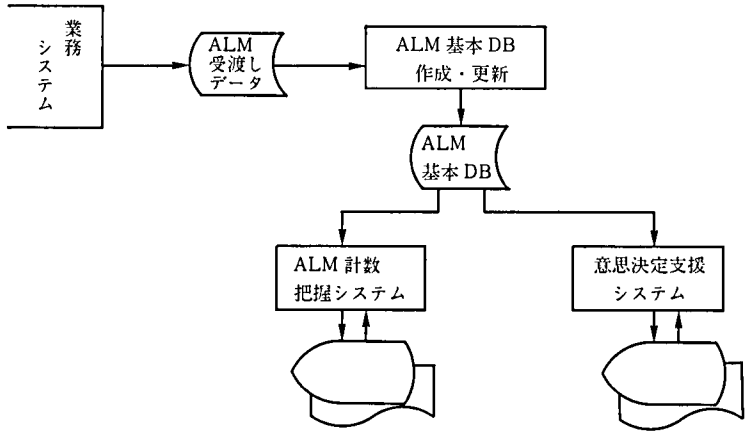


図 2　ALM システム構成
Fig. 2　ALM system structure

2.3 開発期間

ALM システム開発のための調査・検討は，すでに昭和61年から行い，62年4月より開発を始め，約1年半の期間で第一次のシステム開発を終了した（図3）．

2.4 システム規模

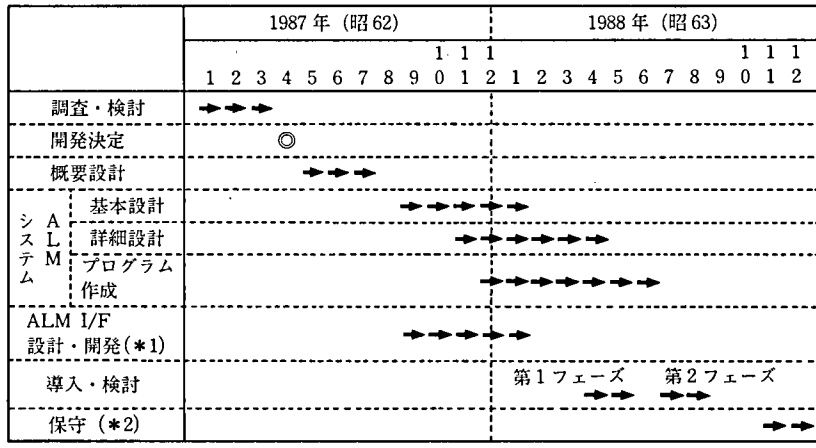
ALM システムのプログラム開発言語別の本数，ステップ数は表1の通りである．開発工数は，約280人月である．

3. システム設計上の留意点

今回のシステム設計に当たっては，次に示す三つの条件を前提とした．

1番目は，ALM システムは当社の販売商品であるということである．商品である以上，

1) ニーズがなければならない．



(*1) ALM システムに入力する受渡しデータ作成システム.
(*2) 保守作業(機能追加)は現在も続いている.

図 3 開発スケジュール

Fig. 3 Schedule of system development

表1 ALM システムプログラム本数およびステップ数

Table 1 The number of programs and steps

言　語	本　数	ステップ数
COBOL	240	10,000
MASM	2	540
MAPPER	250	37,500
SUFICS	52	13,200
計	544	171,240

2) ニーズにマッチしたものでなければならない.

3) 適正な価格である.

4) 採算がとれる.

等の条件を満たすことが必要になる. どんなに良いシステムでも, ニーズに合わなければ使ってもらえない. また, 価格が高くなりすぎても購入しにくくなってしまう.

何を作っていくか, どのような機能を取り揃えておけばよいか, 魅力的な商品とするためには付加価値をどのように付けていけばよいか, それを作るのにどれだけの人・物・金を投入しなければならないのか, どこまで保守を考えておかなければならないか, 等がシステム設計のポイントとなる.

2番目は, ALM システムはパッケージ商品として汎用性を持ったシステムでなければならないということである. 勘定科目等のコード体系はユーザによって当然違ってくる. また, 分析する時の科目の単位（くくり）も変えられるようにしなければならない.

3番目は, ALM のアプリケーションは時代とともに変わっていくものであり, その時代に合ったものに作り替えていく必要があるということである. したがって, 柔軟性と拡張性のあるシステムでなければならない.

この章では，以上の前提条件を踏まえて実際のシステム設計をどのように行ってきたか，システム設計ではどのようなことに注意しなければならないかを，体験に基づいて述べていく．当たり前のことであるがシステムの規模が大きくなるにつれ，システム設計の重要性は増してくる．

システム設計は通常，概要設計と基本設計に分けられる．この区分けは次のように考えている．システムのINPUTとOUTPUTを設計するのが概要設計であり，それを実現するための内部の仕組みを設計するのが基本設計である．別の言い方をするならば，概要設計は外部設計に当たり，基本設計は内部設計に当たる．以下この分け方により，概要設計・基本設計の順で述べていく．

3.1 概要設計での留意事項

ALMシステムの概要設計を行うに当たり，CITIBANKから米国のALMの実態と組織制度面およびシステム面のさまざまなノウハウを吸収し日本の経営風土，自由化の進展状況にあったALMシステムのあるべき姿を検討した．

これを踏まえ，足利銀行と共同で日本の実状に合ったALMシステムの検討および概要設計を約3か月の期間で行った．3か月という比較的短期間で概要設計が終了したのは，事前の調査・研究が十分行われていたためと言える．

ALMシステムの概要設計のポイントは二つあったと考えている．

第1のポイントは，ALMとは何か，それをサポートするシステムはどのようなものであるべきか，の方向付けを最初に行ったことである．当時ALMとは何かということについて二つの見方があった．経営計画・収益管理までを含んだ広い範囲での捉え方と，リスクマネジメントに的を絞った捉え方である．

われわれはALMとは収益管理ではなく，リスクマネジメントを行うものであるという考え方に基づき，システムの検討を行った．リスクとは何か，リスクをどのようにして捉え計量化するか，リスクマネジメントに必要な情報とは何かということを検討した．この最初の方向決定が違っていたならば，当然のことながらシステムの性格・内容が違ったものとなっていたであろう．また，システムの規模もかなり異なっていたと考えられる．

第2のポイントは，必要機能の洗い出しと整理である．概要設計段階では数多くの機能が検討された．考えられる機能を漏れなく取り上げた．次にそれらの整理・体系付けを行った．ここで取り上げられた機能を全部開発したならば450人月以上のパワーを投入する必要があったと見込まれる．開発予算の上限の枠は決まっており，その範囲内に抑さえなければならなかった．重要度・緊急度・開発工数をもとに今回開発分の絞り込みを行った．

10人月，20人月規模のシステム開発であれば，開発工数が1.5倍や2倍になったとしても問題は大きくならない．しかし，200人月の予算でスタートした開発が，2倍の400人月になったとするならば，スケジュール面・採算面・要員面で大きな問題となる．大規模システムになればなるほど，問題とその影響は大きくなる．システムに対するリクエストはいつも多すぎるくらいに出てくる．これをいかに整理し，絞り込んでいくかが重要である．

いずれにしても，この過程では次に示す項目の確認が重要である．

1) 項目（案件）が漏れなくピックアップされていたか．

2) それらについて十分に討議が行われたか．

3) 検討の結果，開発対象外となった項目については，その理由が明確になっているか．双方が納得したか．

4) 検討結果は議事録として残されているか．

3.2 基本設計での留意事項

3.2.1 一般的な考慮点

基本設計を行うに当たって筆者が注意している点は次に示す項目である．

1) シンプルな構造になっているか，いたずらに複雑化していないか．

2) 解りやすいか．設計書を他の人が見て（読んで）そのままで理解できるか．

3) 設計思想が首尾一貫しているか．

これは，データ設計（ファイル設計・レコード設計・テーブル設計・コード設計），プロセス設計（プログラム分割）にも言えることである．

1) 設計はシンプルで解りやすくする……データ設計の方法論や考え方はいろいろあると思うが，ファイル，レコード，項目を設計する場合，異質なものをできるだけ分けるようにした方が望ましいと考えている．たとえば，銘柄の属性管理と残高管理は分けるようにする．レコードを構成する項目に関していえば，一つの項目には一つの役割だけを持たせるようにすることである．

プロセス設計については次のように考える．一つのプロセスには一つの役割だけを持たせる．それにより一つ一つのプログラムは処理が単純になり，サイズ（ステップ数）も小さくなる．さらにプログラム作成も容易になる．

シンプルな設計を行えば，結果として解りやすいシステムができ上がるといえる．設計作業は直進的にはいかない．後戻り・訂正を繰り返しながら進み内容が固まっていく．いくつかの選択枝の中から一つを選んだ場合，あるいは後戻り・訂正を行った場合に，なぜ他を選ばずにそれを選んだか，なぜそのような設計を行ったかを記録しておく必要がある．他の人にとって，そのシステムを理解するのに役立つ資料となり，システムの改訂を行う時に同じ過ちや試行錯誤を避けることができる（図4）．

2) 設計思想が首尾一貫していること……入出力設計（画面・帳表），データ設計，プロセス設計ともに設計は考え方を統一して行う必要がある．とくに，エンドユーザに直接関係する入出力設計は注意が必要である．画面の設計についていえば，タイトル・サブタイトルの表示位置，入力項目と出力項目の識別方法，色の使い方等を統一しておかなければならない．これらの事柄はシステムの本質ではないが，外見の良し悪しがシステムの良し悪しの判断に大きく影響することを認識しておかなければならない．

データ設計，プロセス設計も考え方が統一されているならば，そのシステムは理解しやすくなり，保守もしやすくなる．

3.2.2 設 計 技 法

システム設計（機能分割・プロセス分割）は，T. デマルコ[2]の「構造化分析」を用い，データフロー・ダイアグラムでプロセス構成およびデータの流れを定義している．

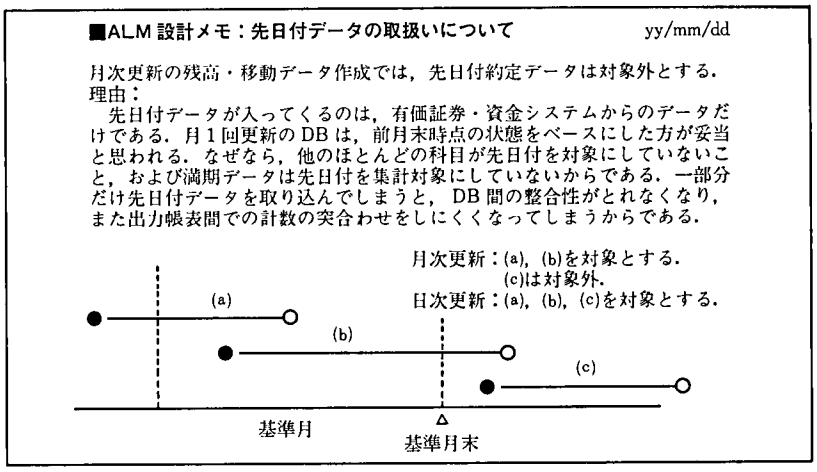


図 4　設計メモの例

Fig. 4　Example memo of system design memo

この技法のメリットとして，次の点をあげることができる．

1)　トップダウン設計に向いている．

2)　多次元の階層的記述ができる．

3)　コンパクトに表現ができる．

データフロー・ダイアグラムの例を図5に示す．

3.2.3　データ設計に当たっての工夫点，注意点

ALMのデータは次の三つに分けられる．

ALM受渡しデータは，勘定系システムから切り出される一件一件の取引データを言う．ALMシステムへの主入力データであり，データの総件数は数百万件となる．

ALM基本データベースは，いろいろな切口で分析ができるようにALM受渡しデータが集計されたデータ群を言う．

分析用データ群は，基本データベースを二次加工して作られたデータであり，MAPPER用データ（ALM計数把握用データ）とSUFICS用データ（シミュレーション用データ）からなる．

それぞれについて工夫点，注意点を以下に述べる．

1)　ALM受渡しデータ……ALM受渡しデータ作成処理は，ALMシステム導入時ユーザが行う．将来の機能拡張に備えて，現在は使用しない項目が受渡しデータ上に多数設けてある．その理由は，ALM受渡しデータ作成処理はできるだけ修正せずに，すなわちALMシステム導入ユーザにあまり負担をかけずに，今後システムのレベルアップができるようにするべきと考えたからである．

2)　ALM基本データベース……ALMシステムのデータ設計で最も苦心したのが，このALM基本データベースの設計である（図6）．科目・残存期間・約定期間・金利更改期間・基準金利・レート等，さまざまな切口での検索や二次加工・分析が容易にできなければならない．ALMのアプリケーションは時代と共に変わっていくと考えられるが，ALM基本データベースは簡単に作り替えるわけにはい

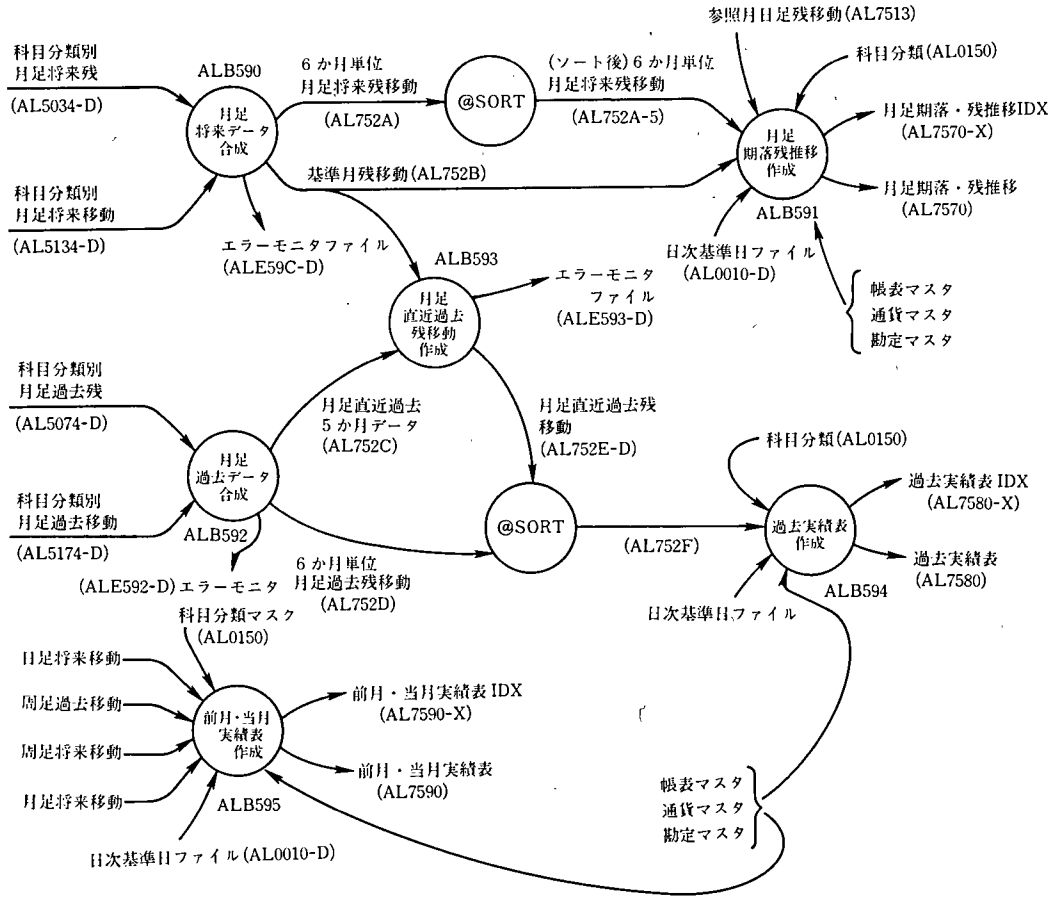


図 5　データフロー・ダイアグラムの例

Fig. 5　Example of data flow diagram

かない．将来のアプリケーションにも応えられるものでなければならない．

基本データベース設計に当たり注意した点は次の通りである．まず，キー項目については将来使用するであろうと考えられるものを漏れなく盛り込むようにした．基本データベースは，時系列構造をした多数のサマリデータ群である．これらのデータはすべて同じキー構造を持たせるようにした．分析処理プログラムでの検索・二次加工が，データが違っても同じように処理できるようにするためである．また，データの圧縮をする場合の指示が統一して行えるからである．

二番目は，一つのデータには原則として一つの用途・役割だけを持たせるようにしたことである．データ構造をシンプルにするためである．

三番目は，科目計・グループ計・全店計等の集計を行う時に，平残およびレートの計算で誤差が出ないように項目の定義をしたことである．

四番目は，分析プログラムが複雑な処理をしなくても済むように，基本データベースを設計したことである．そのぶん ALM 基本データベース作成のサブシステムが複雑でむずかしくなってしまった．しかし，分析プログラム群が簡単にな

ったメリットの方が大きいし，そのメリットはこれからも生きてくることになる．面倒な処理は，分散させずにできるだけ集中して行うようにすべきである．これは，保守の面からもいえることである．

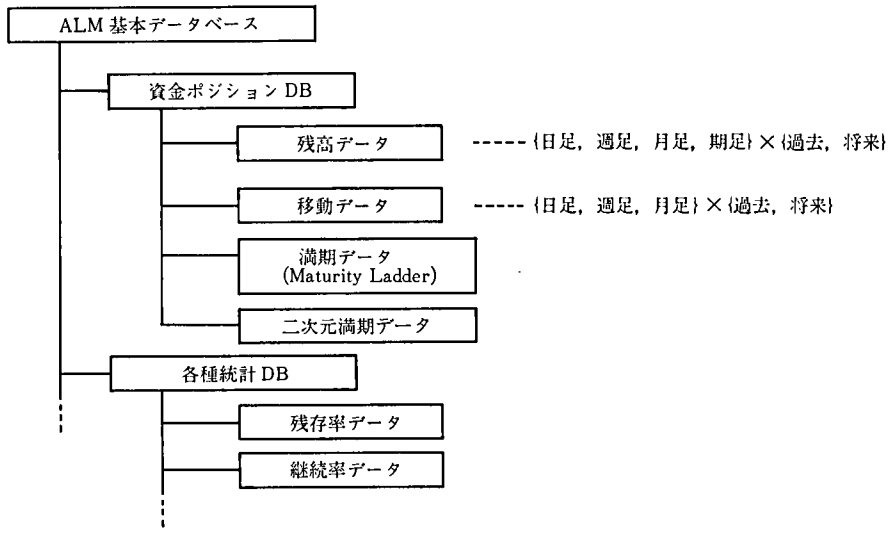


図 6　ALM 基本データベース構成

Fig. 6　structure of ALM system primary database

3)　分析用データ群……ALM 計数把握用データは，帳表出力プログラム（MAPPER RUN）専用のデータとして作られている．これらのデータは，画面からの出力条件（期間・残高区分・利回区分等）に基づき，すぐに編集・出力できるように設計されている．

シミュレーション用データは，基本データベースを圧縮し SUFICS で取り扱えるようにレコード分割・再編集されたデータである．基本データベースのサブセットと考えてよい．SUFICS プログラムの処理効率化のために，これらのデータには SUFICS プログラム専用のキー情報が付加されている．

3.2.4　プロセス設計上の留意点

基本設計作業におけるプロセス分割の最終単位がプログラムとなる．プログラムはできるだけ小さくし，単一の機能だけを持たせるようにすべきである．COBOL のプログラムであれば，500 ステップ以内が適正といえる．MAPPER のプログラムであれば，200 ステップ以内が望ましい．

3.2.5　汎用性の考慮

ALM システムはパッケージソフトウェアであり，多数のユーザに適用できるようにしなければならない．すなわちユーザが異なっても，汎用的に使えるようにしなければならない．

汎用性を持たせるために勘定科目等，ユーザにより違うものはできるだけテーブル形式とし，MAPPER レポート上にその情報を持たせるようにした．MAPPER の会話型機能を用い，このテーブルの登録・変更処理および管理を行う．プログラムは，こ・

のテーブルを参照して処理を行う．

さらに，これらのテーブルの一部の内容をCOBOLの登録集（COPY集）にして，COBOLプログラムは登録集で定義されたデータ名を使用するようにしている．すべてをテーブル化して処理を行うようにすると，プログラムがわかりにくくなってしまうからである（図7）．

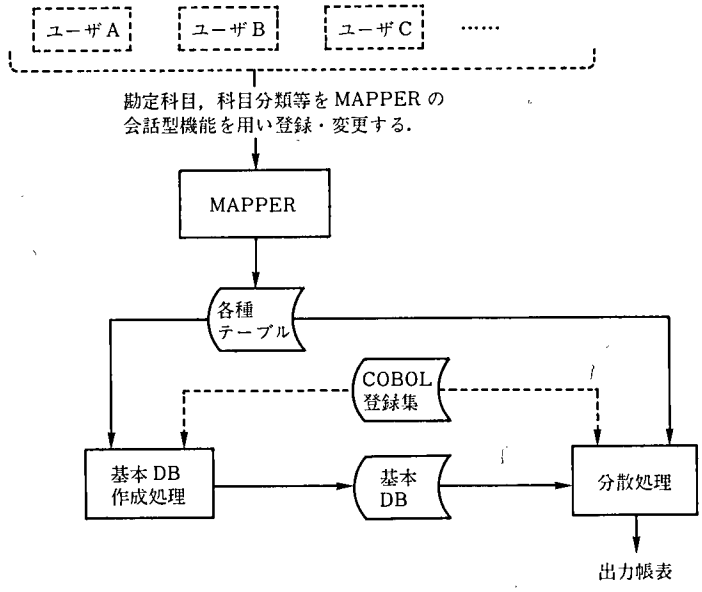


図 7　汎用化の仕組み

Fig. 7　The way to apply this system to many different users

4. プログラム開発上の留意点

この章では，プログラム開発フェーズで何をやらなければならないか，どんなことに注意しなくてはならないかを述べる．ここでは，プログラム開発フェーズをプログラム設計，プログラミング，テストまでの範囲でとらえている．

4.1 プログラミング言語の選択

プログラミング言語を何にするかは，システム設計の段階で決められる．今回のALMシステム開発では，COBOL，MASM，SUFICS，MAPPERの四つの言語を使用した．言語の使い分けは次の通りである．

1) COBOL……ALM受渡しデータからALM基本データベースを作成する処理および二次加工処理では，大量データの集計処理を行う必要がありCOBOLを使用した．

2) MASM……カレンダルーチンをMASM（メタアセンブラ）で作成した．ALM基本データベース作成処理で数千万回という膨大な数の日数計算が行われ，その処理時間がばかにならない．当初，COBOL版カレンダルーチンを使用する予定であったが，日数計算がネックになりALM基本データベース作成に余りにも多くの処理時間がかかることが想定されたため，MASM版高速カレンダルーチンを

作成した.

3) SUFICS……シミュレーションの処理プログラムをSUFICSで作成. SUFICSは豊富な予測手法や時系列分析関数を持っており，表形式のデータの計算命令を簡単に記述できる. またインタプリタ方式なので，プログラミング・テストが簡単に行えるという特徴を持っている.

4) MAPPER……データ入力・ALM計算情報出力・シミュレーション指示はすべて端末より行う. ALMシステムの画面はすべてメニュ方式になっており，誰にでも操作できるように設計されている. このエンドユーザ・インタフェースの処理プログラムをMAPPERで作成した.

どのプログラミング言語を使用するかは，システム開発上の重要な選択となる. 言語によって，プログラム開発（プログラム設計・プログラミング・テスト）の生産性が大きく違ってくる. MAPPERおよびSUFICSは，COBOLに比べて3～4倍の生産性をあげることができる. 処理効率の面では，COBOL等のコンパイラ型言語で作成したプログラムの方がよい.

規模の大きいシステムになるほど，複数のプログラミング言語が使用されることが多くなる. どの言語を何処に適用するか，各言語の開発ボリュームをどのくらいにするかによって，システム開発工数の大小が決まってくるし，またシステム性能にも大きく影響してくる. 処理の内容・処理効率・修正が多いかどうか，開発工数等をよく検討して言語の選択を行うことである.

4.2 スケジュール作成と管理

開発スケジュールをどう立てるか. 開発ボリューム，開発体制と開発要員数，開発マシンの環境によってどのくらいの期間が必要か線引きができる. しかし実際のところは，外的要因で本番実施時期が最初に決められ，それに合わせて開発体制を組み，開発要員を集めることの方が多い. 現在はどこでも要員不足の状況であり，無理なスケジュールにならざるをえない場合も出てくる.

しかし，当然であるが最初から無理なスケジュールは立てるべきではない. スケジュールには余裕を持たせておく必要がある. システム開発の過程では当初予想もしていなかったことが起こるものである. たとえば，パフォーマンスの改善，仕様の変更，サブシステム間のインタフェースの不整合等の設計上の誤り，キーとなる開発要員が健康上やその他の理由で開発プロジェクトから離れてしまう等である. 今回の開発でもテスト段階でパフォーマンスの問題が出て，処理効率改善の仕様変更とプログラムの修正を行った.

このような問題が出た時に，対処できるだけの余裕がスケジュールに加味されていないと，スケジュールの遅れにつながり，ひいては本番実施時期にも影響が出てくる. 無理なスケジュールや余裕のないスケジュールは，開発担当者に精神的な圧力と疲労を与えるものであり，長続きしない. 無理をして頑張れるのもせいぜい半年くらいである. 開発担当責任者は，最初にスケジュールを立てる時にこれらのことを認識し，社内・社外の関係部門との調整を十分に行っておかなければならない.

プログラム開発段階でのスケジュール管理は，週1回の間隔が適切である. 週2回以上であれば，報告作業が大変になり，管理のオーバヘッドも大きくなる. 逆に，チ

ェックが1週間以上あいてしまうと問題の把握と対処が遅れてしまう．

実際の開発作業が当初のスケジュール通りに進んでいれば問題はないが，最初から最後までスケジュール通りに進むことはまずない．

原因としてよくあげられることは，以下の事柄である．

1) 見積りステップ数と実績ステップ数のギャップ

2) 仕様変更が多い．

3) 担当者のスキル不足

見積りの誤りが部分的なものであればまだよいが，全体的なものであれば問題が大きい．開発予算・開発体制・スケジュールを早急に見直し調整しなければならない．

仕様変更についてはその内容にもよるが，システム設計段階であればまだ影響は少ないが，プログラム開発段階ではその影響と作業量は大きくなる．システム設計段階でエンドユーザと仕様の確認を十分行っておき，プログラム開発段階での仕様変更はできるだけしないことが望ましい．

いずれにしても，早めに遅れの原因を調べて対策を講じることである．

4.3 開 発 規 約

プログラム仕様書の作成方法およびプログラムの設計方法は，プログラム言語により異なり，またバッチ・デマンド・リアル等の処理形態によっても違ってくる．

今回の開発では，COBOLで作成したバッチ処理プログラムはジャクソン法，すなわちデータ構造に基づきプログラムを設計する方法を用いた．この方法は設計理論が明確であり，解りやすいプログラム構造を作ることができる．別の言い方をするならば，設計に個人差が出にくいので，他の人が作ったものでも理解しやすい．

MAPPERのプログラムの場合は，処理効率を優先して考えるようにしている．MAPPERのプログラムは作り方によって，処理効率に大きな違いが出てくるので注意が必要である．

コーディングについては，言語別に規約を作成している．

設計規約・コーディング規約等の開発規約は，どのようなシステム開発でも必要なことであるが，規模の大きいシステム開発の場合，徹底するのがむずかしくなってくる．今回の開発では完全に徹底しきれなかった．開発期間が長い場合，最初から最後まで同じメンバが作業するわけではなく，途中で入れ替わることが多い．都度，規約の説明とその徹底についての補足が十分でなかったと反省している．

4.4 プ ロ グ ラ マ

プログラム開発の生産性は個人差が大きい．10倍以上の差があるという報告もある．今回の開発ではこれほどの大きな開きはなかったが，それでも3倍近い差があった．優秀なプログラマが多いにこしたことはないが，彼らにしても最初から現在と同じレベルにあったわけではない．

大規模システムになると，設計・開発・保守のライフサイクルは，最低でも4〜5年以上と長くなってくる．システム開発・保守メンバの技術力向上を長いレンジで考えることが大切である．

5. おわりに

ALMシステム開発を通して，大規模システム開発の考察を行ってきた．ALMシステムの規模は大きい方であるが，今や数百人月程度のシステム開発はざらにあり大規模という感覚が薄れつつある．それだけシステムが大型化してきているといえる．大規模システム開発では，開発の方法論とそれを支えるツールの重要性が増してくる．今回の開発でも，システム設計段階の作業はほとんど手作業であったが，次回以降の開発ではCASEツールの検討を考えている．

参考文献

［1］ B. Williams（静岡銀行ALM研究会訳），上田悦久，日本ユニシス金融マーケティング研究会著，「実践的ALM」，日本ユニシス監修，近代セールス社，1989.

［2］ T. DeMarco，（高梨智弘，黒田純一郎監訳），「構造化分析とシステム仕様」，日経マグロヒル社，1986.

［3］ M. A. Jackson，（島居宏次訳），「構造的プログラム設計の原理」，日本コンピュータ協会，1980.

執筆者紹介　宮田敏光（Toshimitsu Miyata）昭和28年生．50年東京教育大学農学部農学科卒業．同年日本ユニシス(株)入社．金融関係のシステム開発に従事．現在，金融システム本部金融システム一部に所属．

債券トレーディング・システム開発事例

An Example of Development Efforts for a Bonds Trading System

中 尾 晴 夫

要 約 近年，金融・証券業界における情報化戦略の一つの要として位置付けられる，債券トレーディング・システムの開発事例を紹介する．

トレーディング・システムは，商品面（債券・株式・CD・CP・先物・オプション等）の広がりと，機能面（引合・約定・ポジション管理・実績管理・売買シミュレーション・価格分析等）の広がりを要求されるシステムである．今後はさらに東京，ニューヨーク，ロンドンに代表される世界のトレーディング拠点との連動性を持つシステムが要請されていくであろう．

また，当分野におけるシステム構築は，ある時点で収束するものではなく，時代の流れに即応したシステムを連続的に開発し続けることが最大の使命であると認識している．当システムは，トレーディング・システムの全体コンセプトから見れば未だ一部が完成を見た状態ではあるが，連続的システム開発の先兵としての果たす役割りは大きいと考えている．

Abstract The author's intention is to show some thoughts on a large-scale systems development project by describing the actual development of a bonds trading system which is considered to be one of the key factors in information strategy for financial and securities communities.

The trading system itself consists of both a horizontal dimension (standing for financial goods such as bonds, stock, CD, CP, futures, options, etc.) and a vertical dimension (representing functional aspects including bid-offer, trade-done, position management, trade history, trade simulation, price analysis and like). In addition, the outlook clearly indicates the future necessity of systems which have direct linkage between Tokyo, New York, London and other major trading spots in the world.

Our recognition is not that our systems construction efforts in such application areas will come to an end at a certain point of time but that it is our most important mission to continue developing systems capable of responding to user needs which fluctuate as time passes.

Although the system described in this paper is only part of the whole picture of a trading system, the author believes that the role it plays has a significant meaning as a spearhead of continuous development efforts for trading systems.

1. はじめに

当システムは，昨今の金融・証券界における各種システム化動向の中で，最も注目を集めているグローバルトレーディング・システムの一翼を担う債券トレーディング・システムと呼ばれるものである．システムのプロフィールについては，後述の章に譲るが，その開発の背景および位置付けは，企業（当システムは証券会社を対象としている）の有価証券トレーディングにおけるリスク管理（ポジション管理他）を主眼としていること，また企業の情報化戦略の要として早期実現を要請されていること等が最大の特徴として挙げることができる．

周知の通り，債券を中心とする有価証券はその流通銘柄の多様性，連続的な新商品

の発生，世界的な相場観に基づく多種多様な有価証券ビジネスの展開等を背景にシステム化を要請されている．

ここに，われわれシステム開発者は，大きな矛盾に直面することになる．すなわち，連続的多様性を持ち続けるものに対し，ある限定した範囲・機能の枠内でシステム化を実現することは，現在のコンピュータ・システム化における基本的ジレンマである．

したがってシステム開発者の最大の責務は，当面要請されているニーズを選別し優先順位付けしそれらを早期に実現することであり，また稼働後の機能追加・改善を行いやすくするために，柔軟なシステム構造を備えた基本構造を持たせることである．

なぜならば，その時点での実現すべきニーズを最大限盛り込んだとしても，システム開発期間が長期にわたった場合には，システム稼働時におけるニーズが，設計当初の姿から変貌してしまうことが多々あるからである．

本稿は，新日本証券(株)における債券トレーディング・システムの開発事例として，開発作業が一つの区切り（なお，当システムは現在も新機能の追加途上にある）を見た昭和63年度までの範囲内で述べることとする．約37万ステップの規模を持つシステムを，約11か月間という短期間で開発を行ったため，前述した柔構造システムという観点ではいくつかの課題を抱えているものの，実現された業務処理機能はユーザの要求を十分に満たしているものと考えている．今後も時代の流れに則したニーズの選択と，その開発期間の設定がトレーディング・システム構築に際しての重要なマネジメントポイントであることは言うまでもない．

2. システムの概要

本章では，債券トレーディング・システムについて，そのシステムの目的および構造を記述する．

2.1 システムの目的

当システムは，経営支援・業務支援の側面からいくつかの狙いを持って開発された．経営支援とは，現業部門に対し経営戦略を具体的な指標として提示すると共に，その浸透度および達成度を把握することである．また業務支援とは，巨大なトレーディング・ルーム内におけるディーラ，トレーダ，マネジャによるトレーディング業務に対し効率性・正確性を提供すると共に，結果に対するフィードバックを即時に行うことである．

これら二つの狙いを具体的に表現すると以下の通りである．

1) 経営支援

① 指示，報告の迅速化，明確化：経営が決定した方針を債券部門のラインに迅速・明確に反映させると共に，ラインのトレーディング結果を経営が把握する．

② 組織に則した実績管理の向上：商品，組織両面からの実績管理を可能にする．

③ 評価体系の整備：債券部門でのトレーディング結果を人事面，戦略達成度の面に有効な指標でモニタし評価する．

④ リスク把握の迅速化：経営の面から見たリスクを迅速に把握する．

⑤ 経営視点の多様化：従来と異なる新たな視点からトレーディング結果を把握する．

2) 業務支援

① トレーディング業務の効率化，巨大トレーディングルームへの対応：トレーディングルームの巨大化により，'声'を通信手段とした引合業務，約定業務を'画面'中心へと変換し，聞き誤り等のミスを減少させる．

また，約定入力の簡素化を図り，入力ミスを減少させると共に，約定伝票の自動発行を行う．

② 責任，権限の明確化：玉の分別管理や各種警告枠・目標値の設定により，責任と権限範囲を明確にする．

③ リスクの把握：トレーディング業務に係わるリスクを迅速に把握する．

④ 損益，リスクに関する意識の向上：各種の収益性指標，リスク指標を用いた管理による意識向上を計る．

⑤ 顧客動向情報の向上：債券売買に関する顧客の動向把握の向上を計る．

以上の目的を実現するアプリケーション・システムの機能を図示すると図1のようになる．

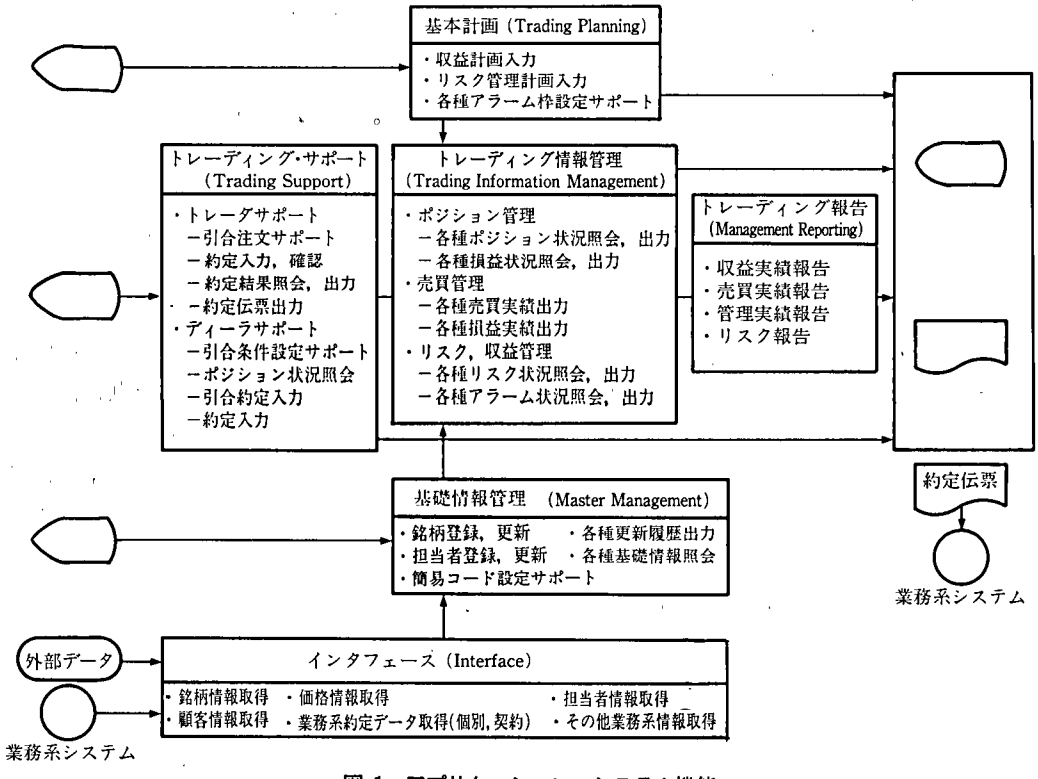


図1 アプリケーション・システム機能

Fig. 1 Application system function

2.2 システム構造

前述した業務機能を実現するためのシステム基本構造は，図2に示す通りである．また，ハードウェア環境は，UNISYS 2200/200 D を2セット専用ホストコンピュー

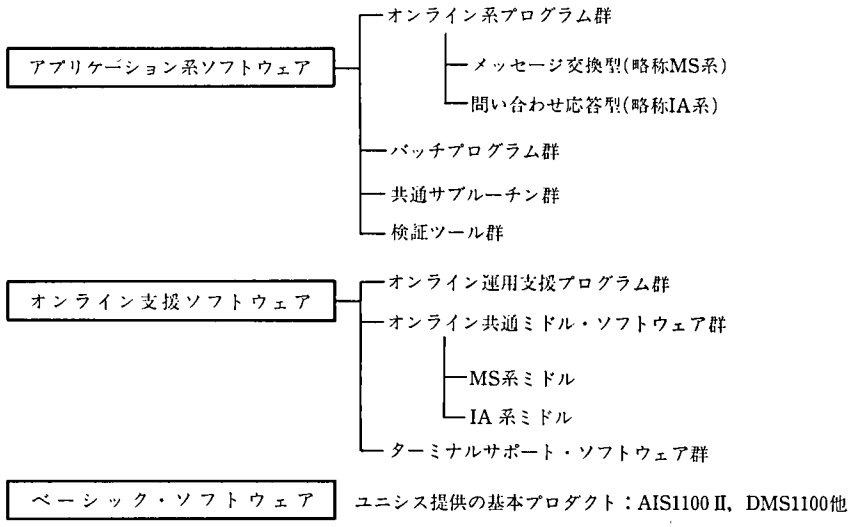


図 2　システム基本構造
Fig. 2　System structure

タとして使用し，トレーディングルーム内では専用端末としてUNISYS-DS 7を各ディーラおよびトレーダに1台ずつ設置している．

このことは，トレーディング・システムには，既存の業務系システムとの連動（各種データ授受が発生）が要請されるものの，業務系システムとは独立したオンライン運用体系をとる必要があったこと，および開発・保守作業がシステム的に独立して行えるように配慮する必要があったことを意味している．

以上のことをオンライン・システムを中心にシステム構造を図示すると図3のようになる．

3. システム開発経緯

冒頭で述べたように，債券トレーディング・システムはまだ開発途上にあるが，ここでは昭和63年度内に実施された開発過程について，いくつかの観点から述べる．

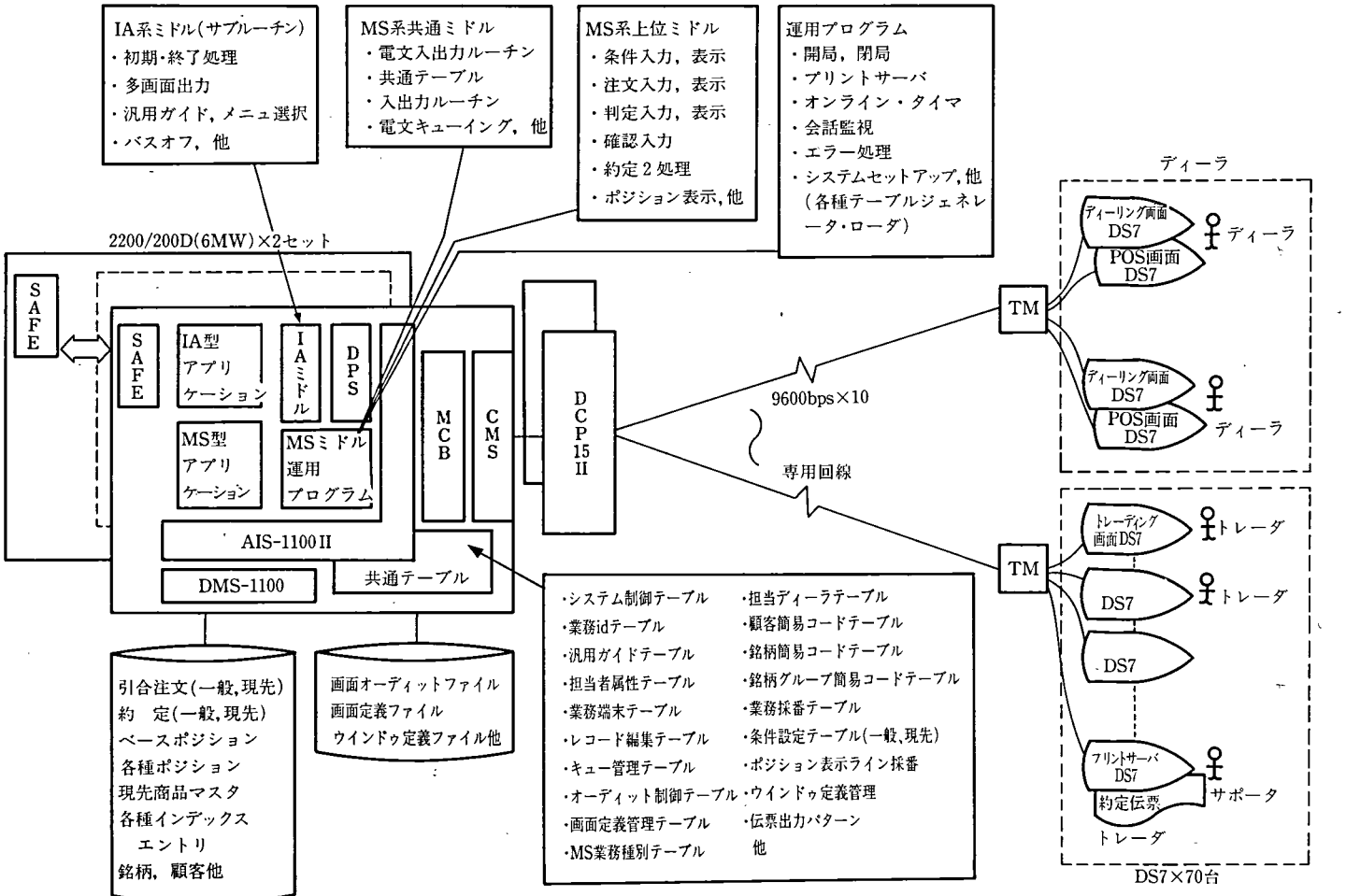
3.1 開発スケジュール

当システムの開発スケジュールは，図4に示すように一般的な組み立てと言える．しかしその中には，短期間で大規模開発を行う場合のいくつかの重要なポイントが設定されている．

なお以下に示すポイント①～④は，図4の開発スケジュール内に表示されている番号に対応している．

① Basic Design（基本設計）を行う以前にシステム概要を十分に把握し，以降の設計を行うための基本方針を策定し最後まで守り通すこと．

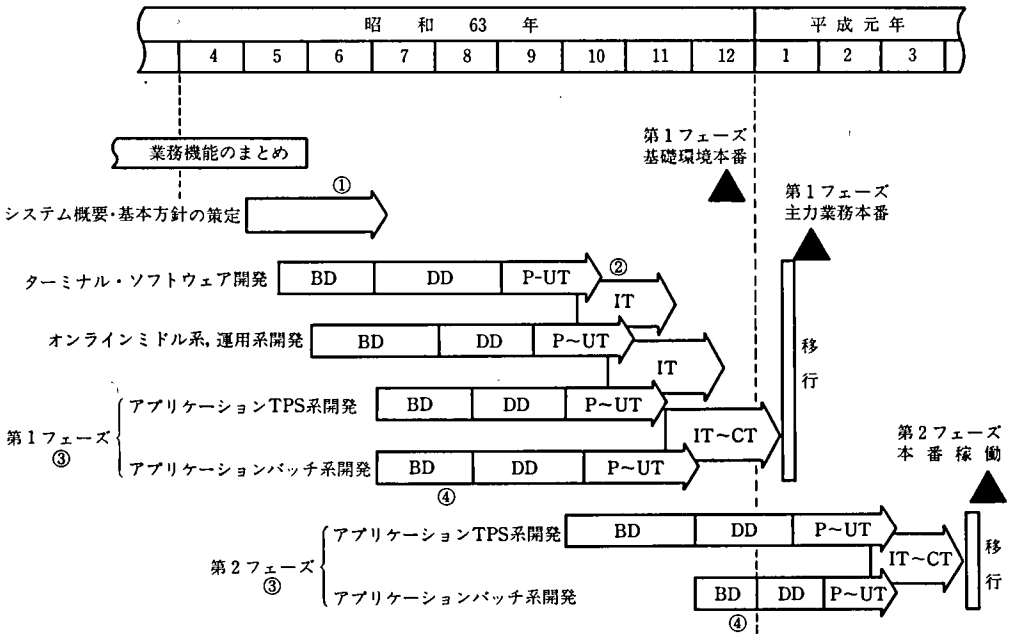
② アプリケーションを支えるミドルソフトウェア・ターミナルソフトウェア系等の基幹部分の仕様を早期に固め，アプリケーションテスト開始前にそのスタビリティを確保しておくこと．



SAFE：ホットスタンバイ構築支援ソフトウェア
DPS：オンライン両面制御ソフトウェア
CMS/MCB：オンライン・メッセージ制御ソフトウェア
AIS-1100II：オンライン・コントロール・ソフトウェア
DMS-1100：データベースマネジメント・システム
TM：ターミナル・マルチプレクサ

図 3　オンライン・システム構造

Fig. 3　Online system structure



BD：Basic Design（基本設計）DD：Degital Design（詳細設計）
P~UT：Programming~Unit Test（プログラミング~単体テスト）
IT~CT：Integration Test~Cycle Test（結合テスト~サイクルテスト）

図 4　開発スケジュール

Fig. 4　Development schedule

③　アプリケーション機能が膨大となるため，エンドユーザへの馴れ等も考慮しリリーズプランをあらかじめ段階的に分けておくこと．

④　Detail Design(詳細設計)/Programming(プログラミング)/Unit Test(単体テスト)等は，極力多数の要員が作業可能とすべくプログラムモジュールの細分化を行うこと．

3.2 開 発 体 制

3.1節にて記述した開発スケジュールを推進するための開発体制を確立する際に，以下のような基本的留意事項を考慮した．

1)　プロジェクト・マネジメント，およびシステム全体の設計思想（アーキテクチャ）を統轄する役割を明確にすること．

2)　開発システムを極力サブシステム化し，各サブシステム単位に責任者を明確にすること．

3)　1サブシステムの開発規模としては，約8万ステップ~12万ステップ規模程度に留めること．

4)　協力会社による支援体制を複数社体制としてワークキャパシティの許容度を上げること．

なお体制の確立は，プロジェクト発足当初は十分ではなく，現実的には前述した開発スケジュール中の基本方針策定作業の中頃であり，また要員面の確保はスケジュール進行中に徐々に整っていった．

3.3 開発システム規模

当システムは，ホスト処理系プログラムに関してはすべてCOBOL言語で記述され，ターミナル・ソフトウェア系プログラムはC言語で記述されている．平成元年3月末時点における開発規模を示すと表1のようになっている．

表1 システム開発規模

Table 1 Total developed program module (AP：アプリケーション)

	サブシステム	言語	プログラム本数	ステップ数 (Kステップ)	備考
第1フェーズ	AP系オンライン・プログラム	COBOL	75	100	
	AP系バッチプログラム	COBOL	102	67	
	オンラインミドル・ソフトウェア他	COBOL	98	70	主としてサブルーチン系
	AP系共通処理	COBOL	78	45	APサブルーチン 検証ツール他
	ターミナル・ソフトウェア	C	35	13	
第2フェーズ	AP系オンライン・プログラム	COBOL	28	45	
	AP系バッチプログラム	COBOL	38	13	
	AP系共通処理	COBOL	24	15	APサブルーチン 検証ツール他
合計			478	368	

この表から，とくにアプリケーション系プログラムの特性を分析してみると以下の特徴がある．

1) オンライン・プログラム：1本当たりの平均ステップ数は，約1,400ステップ（実際には500ステップくらいのものから，4,000ステップくらいのものも存在する）

2) バッチプログラム：1本当たりの平均ステップ数は約570ステップ

オンラインプログラムは，画面の入出力を基本としてプログラムが起動されるため，エンドユーザニーズとプログラムの対応が比較的同期しやすいことを示しており，バッチプログラムは一般的にエンドユーザニーズとは直接同期しない（最も同期するのは帳票編集プログラム）ため，設計者の意図，処理効率/障害対策との関連等，システム開発者の考え方にそってプログラムが分割される傾向が強いことを示している．また，短期間の開発が要請されたためバッチプログラム，サブルーチン系等に関しては意図的にプログラムを細分化し，多数の要員に作業が分散できるよう配慮した．

4. 技術的開発ポイント

前述したスケジュールが示すように，短期間の大規模開発が要請されたために設計開始前に下記に示すような基本方針を設定し，その範囲内で業務ニーズを実現させることを遵守した．

もちろん，このことは開発側の一方的論理ではなく，ユーザによる十分な理解を得られたものでなければならない．

基本方針の設定内容は，次に示す9項目が挙げられる．

1) 当社提供の標準ソフトウェアプロダクトを十分活用すること．

2) アプリケーション処理と共通モジュール（オンラインミドル・ソフトウェア等）を明確に分離すること．

3) データベース構造を簡素化すること．

4) オンライン運用，障害対策等の実現機能を絞り込むこと．

5) ターミナル・ソフトウェア開発の主眼は端末操作性の向上を狙いとすること．

6) 使用に耐え得るオンライン応答時間を保証するためのマスタファイル更新方式を設定すること．

7) 移行処理と日次処理の処理プログラムを共通化すること．

8) オンライン処理体系（メッセージ交換型(MS 型)・問い合わせ応答型(IA 型)）を標準化すること．

上述した基本方針の中から，とくに当システム固有の処理方式である 6)，7)，8)について若干述べることにする．

4.1 オンライン応答時間を保証するためのマスタファイル更新方式

オンライン処理中は，トランザクション・データ発生の都度マスタファイル（例：ポジションデータベース等）を更新し，常に最新の状態を保つことが必要であるが，トレーディング業務においては過去のデータに遡り，ファイルの再更新を必要とするトランザクション・データが多々発生する．

したがって，限られたコンピュータリソースにて使用に耐え得るオンライン応答時間を保証するための処理方式を決定しなければならない．

たとえば，その一例として有価証券のポジションを管理する上で，重要なポイントである簿価単価の把握における処理方式について述べてみる．

一般に簿価単価の計算方式として移動平均法・総平均法があり，その概略についてまず述べておく．

〈移動平均法〉

$$\text{買約定発生時：簿価単価}=\frac{\text{前簿価金額}+\text{買金額}}{\text{前額面}+\text{買額面}}\times 100$$

$$(\text{簿価金額}=\text{前簿価金額}+\text{買金額})$$

$$\text{売約定発生時：簿価単価}=\text{前簿価単価}$$

$$\left(\text{簿価金額}=\text{前簿価金額}-\frac{\text{売額面}\times\text{前簿価単価}}{100}\right)$$

すなわち，移動平均法による簿価単価の求め方は，約定発生都度その額面による加重平均を求める考え方である．

〈総平均法〉

$$\text{簿価単価}=\frac{\text{前日簿価金額}+\text{本日買金額計}}{\text{前日額面}+\text{本日買額面計}}\times 100$$

$$\left(\text{簿価金額}=\text{前日簿価金額}+\text{本日買金額計}-\frac{\text{本日売金額計}\times\text{簿価単価}}{100}\right)$$

すなわち総平均法による簿価単価の求め方は，締めた時点ですべての約定の合計値により求めるため，約定の発生順序は問わない考え方である．

今回のトレーディング・システムにおいては，約定の発生都度リアルタイムにポジションを把握することをシステムの基本的な狙いとしているため，簿価単価の計算方

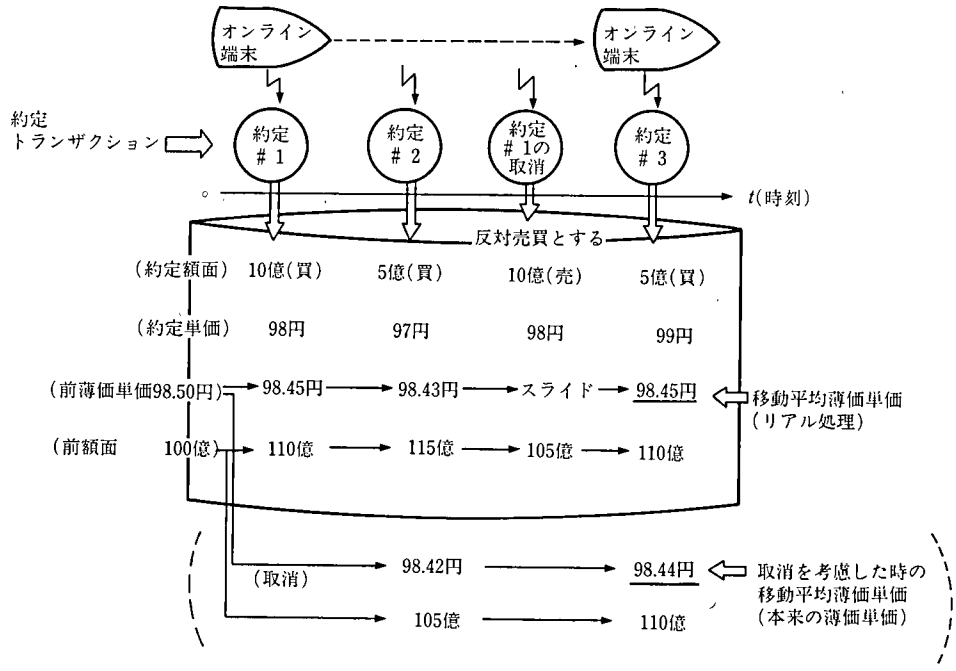


図 5 移動平均法とデータベース更新

Fig. 5 Moving average method and database updation

法として移動平均法を採用している．

したがって，オンライン中の約定処理においては約定の発生都度ポジション・マスタ中の簿価単価の移動平均を算出し，データベース中にファイリングしている．

図5のように以前約定されたデータに対し，取消・訂正等が行われるとリアル処理中の直近移動平均簿価と，本来求める移動平均簿価との間に誤差を生じてしまうことになる．オンライン処理において遂次，本来の移動平均簿価を求めるためには取消・訂正等が行われる都度オリジナル約定レコードの1件前に遡り，直近移動平均の再計算を行わなければならない．

この処理は，現在のコンピュータ技術においてはかなりの負荷を必要とするものである．また，業務運用上数日以前の約定レコードに対し取消・訂正を許すシステム機能を要求された場合には，そのコンピュータ処理負荷は膨大になる．

したがって当システムにおける処理方式の基本方針は，オンライン処理では図5のように発生都度算出（仮計算）し，取消・訂正に伴う誤差の補正はオンラインクローズ後のバッチ処理において，すべての移動平均簿価の再計算を行うこととしている．翌日のオンライン開始時点には，すべての誤差が補正された正しいポジション・データベースをエンドユーザに提供することができるわけである．この点を簡単に図示すると図6のようになる．

以上，オンライン処理およびバッチ処理におけるポジション・マスタの基本的な更新方式について述べた．この点は，当システム開発の最も重要な技術ポイントの一つに数えられる．

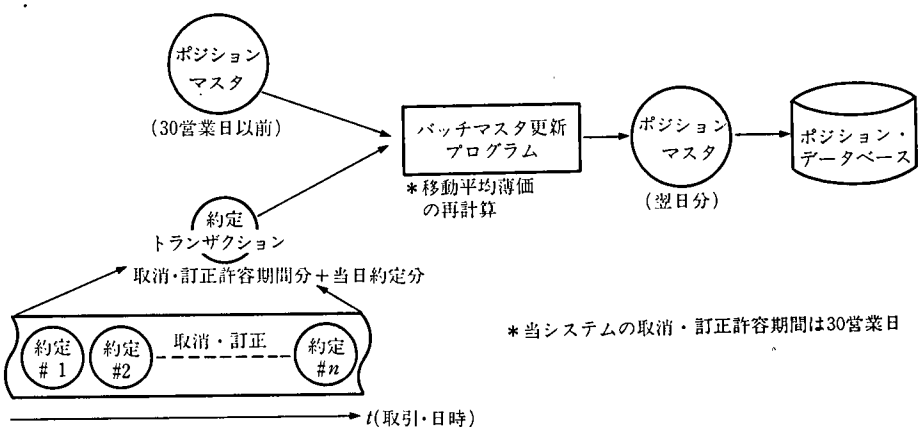


図 6　日次ファイル更新方式

Fig. 6　Method of daily master file updation

4.2　移行処理と日次処理の共通化

一般に新システム稼働時には一過性の処理ではあるが，必ずシステム移行処理を必要とする．したがって，システム開発全体の作業項目の中に移行処理専用のプログラム開発が必要となり，場合によってはかなりの作業工数を要することがある．今回のシステムも例外ではなく移行処理が必須であったが，短期間による開発を要請されていたため移行処理用のプログラム開発工数の削減が重要なポイントとなった．

そこで，移行マスタファイル（例：ポジションマスタ等）の作成プログラムを，極力日次バッチ処理で稼働させるもので代行できるように移行方式を設定した．代表的なケースとしては，4.1節で述べたポジション・マスタ更新プログラムを移行プログラムとして稼働させ，本番初日のポジション・データベースを作成したことが挙げられる．この点を簡単に図示すると図7のようになる．

もちろん，移行専用プログラムが皆無というわけにはいかず，若干開発したことも事実である．主力ファイル作成処理が日次処理方式と同一のため，日次処理の安定化がそのまま移行処理の信頼性を高めることになり，かつ全体の開発工数削減にも繋るため当方針の設定は非常に効果的であったと考えている．

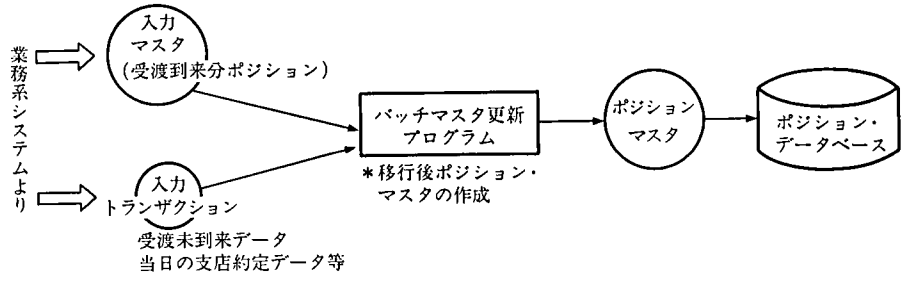


図 7　移行ファイル作成方式

Fig. 7　Method of master file conversion

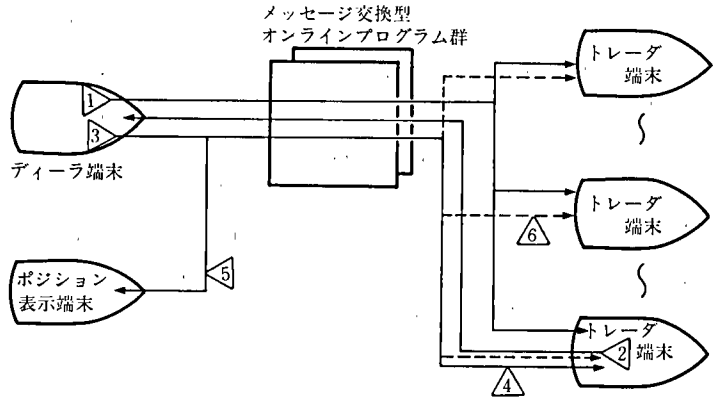
4.3 オンライン処理体系

2章のシステム構造にて述べたように当システムのオンライン処理プログラムは，エンドユーザによる操作性・業務処理要求等の観点から大きく二つに分類されるが，その考え方はおおむね以下の通りである．

	メッセージ交換型 (MS系)	問い合わせ応答型 (IA系)
入力項目	少	多 (少いものも可)
出力項目	比較的少	多 (少いものも可)
操作性	数タッチで可能	メニュ方式
レスポンスタイム	小	大のものも許す
非同期電文の入出力	可	不可
1：mの端末間送受信	可	不可

図8，9にメッセージ交換型，および問い合わせ応答型オンライン処理の例を示す．

なお，メッセージ交換型オンライン・システムは，当システムの目的の一つであるトレーディング業務の効率化を最大の狙いとしており，システム動作は非常に複雑であるが，エンドユーザ操作は極めて簡素化されている．このため，ホストと端末のソフトウェアは相互に一体で動作するように設計されており，限定されたアプリケーシ



1 ディーラによるビットオファ条件の提示

2 トレーダによる引合注文の入力

3 上記引合注文に対する判断入力

4 ディーラによる判定結果の通知

5 ディーラ向けポジション表示の更新(約定成立時)

6 トレーダ向けビットオファ条件の更新(約定成立時)

図 8 メッセージ交換型オンライン処理例

Fig. 8 Example of message switching online transaction

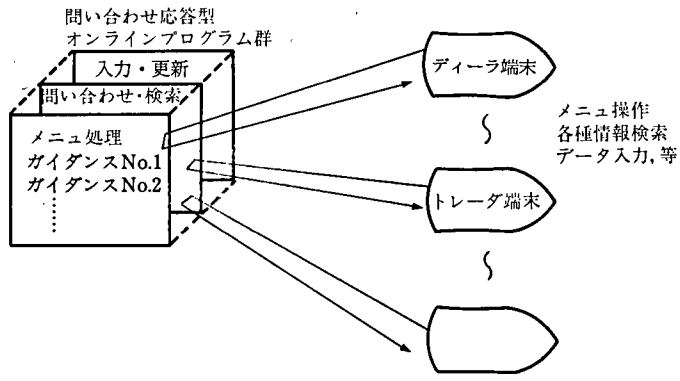


図 9　問い合わせ応答型オンライン処理
Fig. 9　Form of inquiry answer online transaction

ョン処理にて利用されている．

エンドユーザの各端末(DS7)は，メッセージ交換型と問い合わせ応答型の両者を同時に使用することができるため，オンライン処理要求の実現に当たってはいずれの形態にて構築するかをあらかじめ確定し，実運用上のニーズ調整を行うことがポイントとなってくる．

5. プロジェクト・マネジメント

大規模開発においては，プロジェクト・マネジメントが最も重要であることは言うまでもない．今回のシステム開発に際しては，特別なマネジメント技法なり開発技法は用いていない．したがって，プロジェクト・マネジメントは以下に示す当たり前のことを確実に実施することを基本姿勢としてプロジェクト構成員に徹底を図った．

1)　ユーザと当社の役割分担の明確化
2)　開発サブシステム分けと担当リーダの設定
3)　きめ細かな進捗管理と問題点の早期発見
4)　担当要員（協力会社のメンバも含め）の個人的状況の把握
5)　仕様調整，変更等の伝達ルート，手段の明確化
6)　開発リソース（ホストマシン，端末等）の十分な確保
7)　大項目から小項目に至るまで常に作業上の優先順位の意識
8)　開発プロジェクト外部門(例：債券部,運用部門等)への早めな事前調整の実施
9)　具体的かつ詳細な結合テスト，総合テストプランに基づく作業の実施，等

当たり前のことを箇条書きすることは容易なことであるが，日々の開発過程の中で完璧に実践することは至難の技である．

当開発作業も順調に工程が進んだわけではなく，局面に応じた各種の難問が山積みしていたが，前述したポイントを徹底することにより，解決できた問題も数多くあった．

しかし，システム開発計画の成否を左右する最大の課題は本来システム化すべき機能を定義・把握することであろう．一般に開発初期の段階で機能定義を完全に実施す

ることは困難である．しかしある程度の確かさでつかむことができれば，それ以降の開発工程は，作業量と期間および開発パワー(要員)との見合で，その計画の成否は自ずと決まってくる．

システム化機能を定義するプロセスは，エンドユーザ(人)が実現したいニーズを表現し，それを理解(人)し，適切なハードウェア/ソフトウェアの下でシステム・イメージを描き(人)，設計書と呼ばれる媒体に記述(人)することである．まさに人と人のコミュニケーションの繰り返しが基本動作となって，設計作業が行われていくことが基本的過程である．

さらに，開発工程は一般に詳細設計・プログラミング・単体テスト・結合テスト・総合テストへと進んでいくわけであるが，各工程においても書面(設計書等)および会話によってプロジェクト構成員がコミュニケーションを図ることによって，システム化機能がソフトウェア化されその精度が高められていく．したがって初期段階における機能把握の誤解は，システム開発規模が大きくなればなるほど，開発工程が進めば進むほど，その軌道修正には多大な時間と労力を費すことになる．今回の開発作業においては，人と人とのコミュニケーションおよび初期段階における機能把握が，比較的スムーズに行われたことがプロジェクト成功のための大きな要素であったことは言うまでもない．

しかし，当開発作業の最大の難問は，開発規模の大きさに比してその期間が短かった点であろう．この難問を乗り越えた最大の要素は，システム技術力もさることながら，プロジェクト構成員相互の信頼関係とシステム完成へ向けての力強い情熱が根底のエネルギーになっていたことを最後に付け加えておきたい．

6. おわりに

当システム開発が，開始された初期の頃は，当社のプロジェクト構成要員はわずか2名であったが，開発が進行するにつれ63年度のピーク時には，協力会社の構成要員を含め総勢50数名に及ぶプロジェクトに成長していたのである．したがって，全構成要員が一箇所に集まって作業を行う場所も確保できず，四箇所のビルに分散し開発作業を行ってきた．振り返ってみると離れた場所に居る開発部隊間のコミュニケーション，意思統一を図ることにかなりの神経を払ってきたように思える．

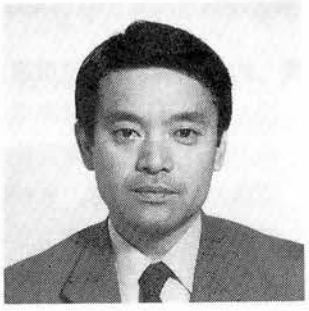
冒頭にも述べたように当システムは現在も発展途上にあり，分散した作業場所をベースに約14万ステップ規模の追加システムの開発が行われており，さらに本稿が出版される頃には次の追加システムの開発が行われているであろう．

金融・証券界における情報武装は留まることを知らない勢いで進行中であり，われわれはその渦中に存在しシステム開発に従事していることは，その業界のトレンドの一端を肌で感じることができるわけである．システム開発作業を，納期・品質・生産性等の月並みな指標で見るだけではなく，開発作業を通し業界のトレンドをキャッチアップすることにより，これからの当社の歩むべき方向にもいくつかの示唆を与えてくれるものと考えている．さらには業界のシステム化動向に対し，システムインテグレータの立場からアクティブな提言・提案を行っていくことも今後の当社の使命であると考えている．

最後に，当債券トレーディング・システムの開発作業を通し多大なる御指導，御協力を賜りました新日本証券(株)ならびに新日本コンピュータシステム(株)さらに，非常に厳しいスケジュール下にもかかわらず御尽力戴いた協力会社各位の方々に対し，改めて謝意を表したい．

執筆者紹介　中　尾　晴　夫(Haruo Nakao)

昭和25年生．昭和47年慶応義塾大学商学部卒業．同年，日本ユニシス(株)入社．52年より証券関連のSEサービス，オンラインシステム開発，アプリケーションシステム開発等に従事．現在，金融システム本部・証券システム一部に所属．



九州地区農協信用事業情報系システム

Financial Information System for Credit Business at Kyushu-region Nokyo (Agricultural Cooperative)

前山泰治, 平嶋孝正

要　約　近年，農協信用事業は他金融機関との競争の激化に伴い，資金量の伸び悩みや運用利回りの低下等，経営環境は厳しい局面に直面することになり，この一つの打開策として，農協の渉外力・経営管理機能の強化が，農協信用事業の取り組むべき重要な課題として共通の認識となっている．九州地区においても，これらの具体的な解決策として，共同センタとしての九州地区農協オンラインセンタ勘定系で蓄積したデータとネットワークを活用し，早急に情報系システムを構築することが必要であるとの認識のもとに，平成元年6月にシステムを稼働させた．

ここでは，比較的大きなシステムとなった情報系システムを，1年余りの短期間で開発できた要因を中心に，共同センタとしての情報系システムの概要と仕組み，および情報系システムの開発工程について紹介する．

Abstract　With Nokyo's (agricultural cooperatives') competition with other financial institutions becoming intensified in recent years, their credit business has been faced with the strict environment which hampers an increased volume of funds and reduces fund-operating yields. It is now commonly recognized that, as one of the solutions, Nokyo must take it for an essential pursuit to improve their relations with the public and enhance their business management.

In the Kyushu region as well, a new system started to operate as a specific solution in June 1989 under the recognition that it was necessary to build a financial information system without any delay, taking advantage of the data and networks amassed and designed for the accounting system at the Kyushu-area Nokyo on-line computing facility which now serves as a joint-use computer center.

This paper discusses the outline and structure of the developed information system working as a joint-use host and its development process, with focus placed on factors which made it possible to finish developing the comparatively large-scale financial information system in as short a period as a little over one year.

1. はじめに

九州地区農協信用事業のオンラインシステムは，昭和55年10月全国系統初の共同オンラインシステムとして稼働し，現在まで各種貯金・為替・貸出金等の既存商品のシステム化をほぼ終えるとともに，全国農協貯金ネットサービス・全銀データ通信への加盟，また地域CD提携の実現等のネットワーク網の整備拡大を図り，現在参加店舗数約2,300店舗，マスタ数約2,000万件の広域オンラインシステムを完成するに至っている．

このように，共同化により比較的規模が小さい九州地区においても多額の投資・要員を要するオンラインシステムを柔軟かつ効率的に構築ができ，また事務処理の標準

化・効率化等により利用者サービスの向上が図られた．他金融機関との競争力確保の土台ができたといえる現在，今後ますます多様化・複雑化する利用者ニーズや金融機関競争に各県および九オンが，どのように対応すべきか新たな課題に取り組んでいる．

2. 情報系システム開発の背景と基本方針

本来，農協独自・県独自であるべき情報系システムを，なぜ共同センタである九オンが構築したか，その経緯と構築の基本方針は次の通りである．

2.1 情報系システム開発に至った経緯

九オンにおける情報系システムの取り組みについては，当初情報管理の農協特性・県特性および共同センタの特性と現状から各県の構想に委ねるとしていたが，金融環境の変化の中で，他金融機関との競争力強化や組合員，地域住民へのサービス提供が要求されてきた．

しかし，その対応としての情報系システムに対する各県の考え方や，取り組みの現状にはかなりの格差があり，この県間の格差は近い将来縮まるという方向よりも，むしろ拡大する方向に進む可能性が強いと思われた．

このことは，九州地区農協信用事業の今後の展開に大きな影響も考えられ，共同センタとしての九オンがその勘定系システムとネットワークを活用し，情報系システムの構築を行うことで九州７県にとってより効率的となる．また，情報系システムにおける県間格差拡大の阻止あるいは短縮が期待できれば，九州地区農協信用事業の今後の全体的発展に寄与できる．以上の観点から，九オンでのシステム開発に取り組んだ．

2.2 情報系システムの基本方針

共同センタである九オンの情報系システム構築において，どこまでシステム化が可能か，また九オン情報系システムがどうあるべきかについて，次のような点をねらいとしている（図１）．

1) 九州７県に利用価値のある情報系システム……九オン情報系システムは，情報系システムの取り組みが進んでいる県と，具体的に未検討の県の双方から，情報系システムに求められている機能をできるだけ具備し，九州７県のどこの県にとっても利用価値のある情報系システムとなることをめざす．

2) 標準システムとしての情報系システム……県独自の情報を付加し，県固有のニーズにマッチした信用事業情報管理をめざしたシステムを九オンで開発することは無理があり，九オンで開発する情報系システムは最低限度必要な情報を展開する標準システムとして開発する．よって，県独自の情報を具備したシステムは各県で構築することとする．

3) 勘定系システム負荷の軽減……勘定系システムについてはオンライン稼働以来，数多くの改善要望が出され，とくに還元帳票や照会に対するものが多く，集約すると次のことが言える．

① 必要な情報に手間をかけず，二次，三次加工することにより目的にかなった資料を手に入れたい．

② 出来上がった資料から必要な時に，必要な情報を得たい．

これらの要求に対応するには，システムの共同利用という形態のため一定の限

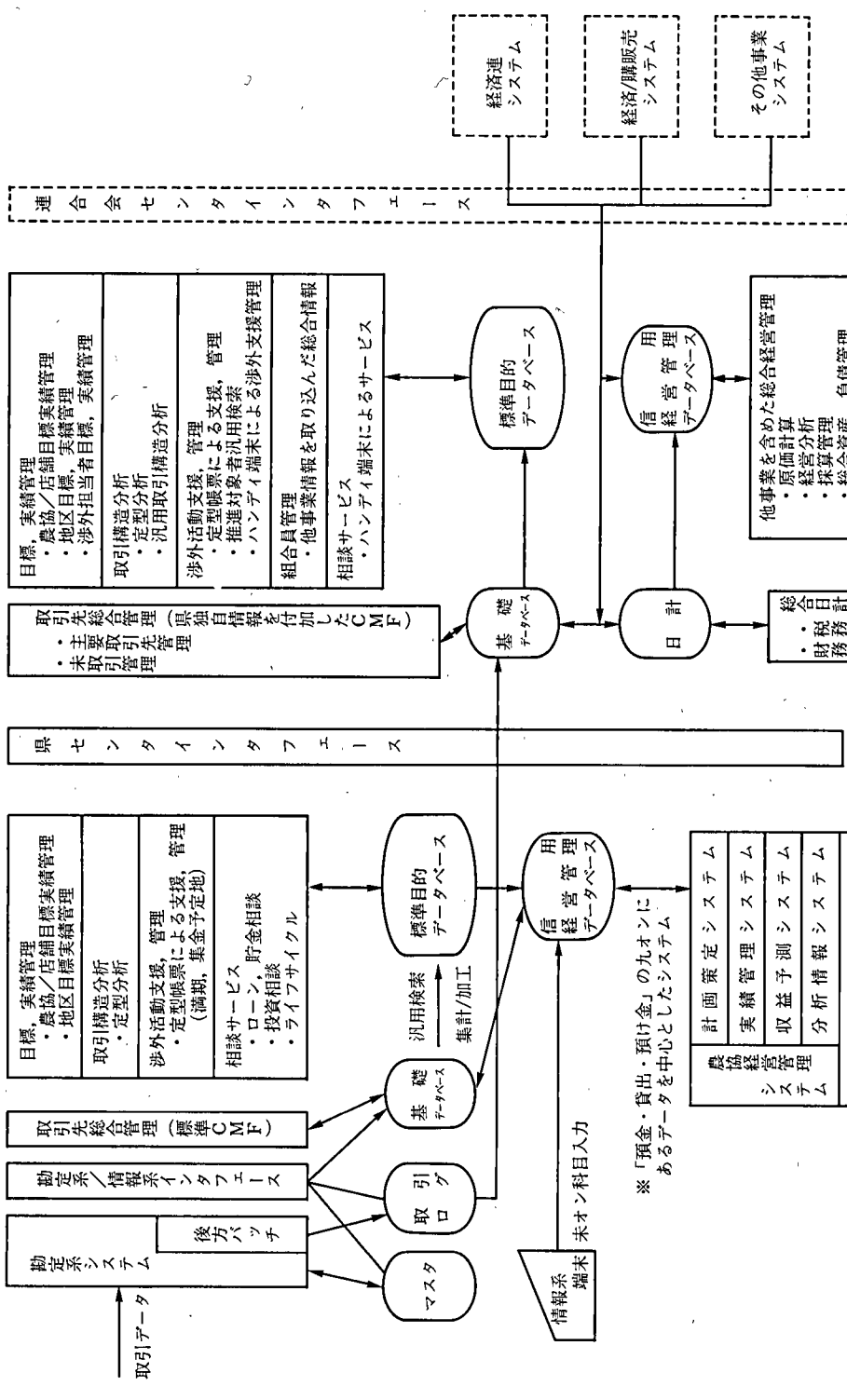


図1 九オン情報系の範囲

Fig. 1 Scope of QON information system

度があり，対応しようとした場合に勘定系負荷がますます増大する．そこで今後，金融機関の品揃え競争のもと少量多種の新商品開発が増え続ける中で，勘定系システムをできるだけシンプルで効率的かつ安全性の高いものとすることが大きな課題となる．勘定系に直接的に影響しないで情報の管理を行い，勘定系システムと機能を分担できる情報系システムが望まれる．

4) 県総合情報システムとの機能分担……九オンは，九州地区農協信用事業の共同センタであり情報系システムにおいても，信用事業に関する情報系システムを構築する．

一方，県総合情報センタ(仮称)は信用事業を含んだ農協総合事業としての情報系システムを構築する．

この場合，九オンで構築する信用事業に関する情報系システムの部分については，県総合情報センタで構築する必要はない．そのため県総合情報センタは九オンとインタフェースをとって，九オン情報系システムが持つ情報を県総合情報システムの中に取り込む．さらに，九オン情報系システムが構築できない信用事業に関する情報系システムを手当てすることになる．

2.3 システムの設計方針

標準システムとして構築する九オン情報系システムの設計方針は次の通りである．

1) 情報系システムのノウハウを取得し，また開発期間の短縮をはかるため，鹿児島県信連において九オン勘定系データをもとに先行構築されたKINGS（鹿児島県信用事業情報システム）を，系統信用事業ソフトウェア流通制度に基づき導入し，参考にして構築する．

2) 開発効率をよくするための開発ツールとして，第4世代言語 (MAPPER)，意思決定支援システム(SUFICS)を導入し，さらに仕組みのためのソフトウェアとしてDIP*を導入する．

3) 情報の検索，出力および二次加工はすべて情報系端末で行いセンタ出力は行わない．

4) 環境ニーズの変化に対して柔軟に対応できるように，スクラップ・アンド・ビルド方式を適用する．

5) 多種多様の目的に利用できる，CMF（顧客情報管理データベース）による情報系基礎データベースを構築する．

6) 情報系基礎データベースの情報は，常に最新情報を蓄積するために日次更新，またはディレード更新を行う．

3. 情報系システム開発における組織構成

九オン情報系システムは，共同開発にもかかわらず早期稼働することができた．このことは，早期開発ニーズに対応するための組織，開発体制および各県と九オンが一体となって取り組んだことによる（図2）．

---

* DIP: Dynamic Database Interface Package, 金融機関情報系システムの構築と運用を支援するフレームワーク・ソフトウェア．

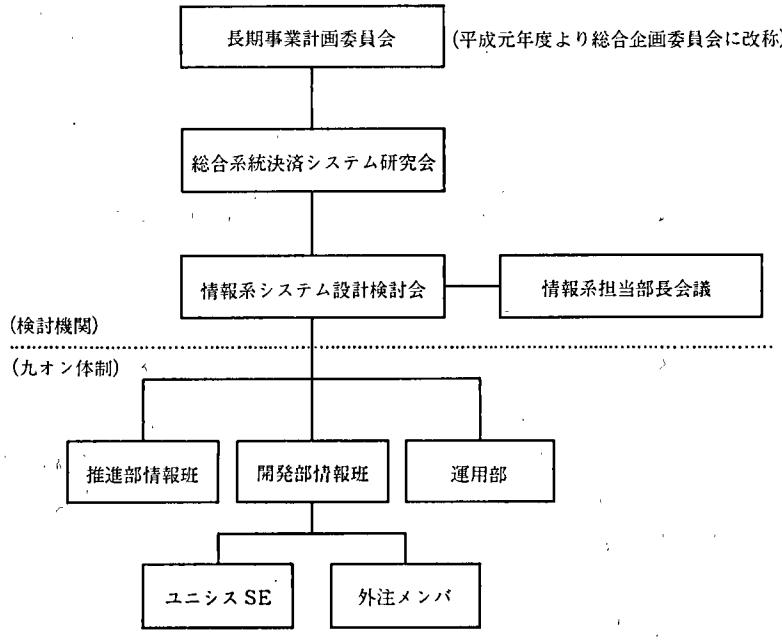


図2 情報系システム開発における組織構成

Fig. 2 Organization of QONdevelopment team

3.1 方針の決定

1) 長期事業計画委員会……九州地区農協信用事業システム化の業務企画の一つとして，共同センタにおける情報系システムの効率的な構築はいかにあるべきか，専門的な研究会を設置して基本的方向付けを検討した．構成メンバは，各県信連業務企画担当部門の代表者(参事，部長クラス)，農林中央金庫福岡支店次長，九オン専務取締役および各部長である．

2) 総合系統決済システム研究会……長期事業計画委員会の決定を受けて発足し，下記の項目について研究，検討を行い最終レポートとして答申した．このレポートが九オン情報系システムの基本方針となった．

構成メンバは，各県信連情報系担当代表者(課長クラス)，農林中央金庫福岡支店代表者，および九オン事務局（各課課長）である．

検討項目は以下の通りである．

- 各県の情報系システム構築に関するニーズの把握，取り組むべき課題
- 情報系システムの基本的方向付け
- 情報系システムの概要
- 情報系システム開発スケジュール
- 勘定系システムとの関連整理
- 情報系システム開発における経費負担の考え方
- 他事業システムとのインタフェース
- その他情報系システム構築に関すること

3.2 システムの検討

1) 情報系システム設計検討会……総合系統決済システム研究会で研究，検討された九オン情報系システムの基本的方向を踏まえ，構築するシステムの詳細な内容等を検討する．システム設計の具体的方向を検討するとともに，システムのスムーズな運営と効率的な利用を検討する．構成メンバは，各県信連の電算部門および利用推進部門の課長または担当，農林中央金庫福岡支店および九オン事務局である．

検討項目は以下の通りである．

- 標準システムの範囲
- データベース化の範囲
- 情報系システムの改善要望の整理
- システムの利用，推進方法，定着化の検討
- その他システム開発上の問題点の整理

2) 情報系担当部長会議……九オン情報系システム開発の基本事項，各県間の調整を要する事項等について協議決定するとともに，情報系システムの効率的な利用，推進を図るための定着化対応を検討する．構成メンバは，各県信連情報系システム総括部署の担当部長，農林中央金庫福岡支店，九オン専務取締役および各部長，次長である．

4. 情報系システムの概要

4.1 システム体系

九オンの情報系システムは，次の6システムを基幹とし，それぞれのサブシステムから成っている（図3）．

1) 基本情報システム
2) 農協営業支援システム
3) 農協経営管理システム
4) 県下取引分析システム
5) 信連経営管理システム
6) 窓口相談業務システム

4.2 サブシステム概要

4.2.1 基本情報システム

1) 顧客情報管理……取引先に関する情報（家族構成・職業・勤務先等），および取引状況（貯金・貸出金の口座や取引実績等）をCMFにより一元管理する情報系システムの基礎となるシステムである．このシステムを利用することにより，個人または世帯の貯金・貸出金の残高や，自動振替契約状況を一目で見ることができる．

2) テーブル情報管理……農協名・店舗名・世帯数・組合員数・職員数等の農協の基本情報を管理する．

4.2.2 農協営業支援システム

農協営業支援システムは，貯金・貸出金等の主要勘定科目や家計メイン化種目等の

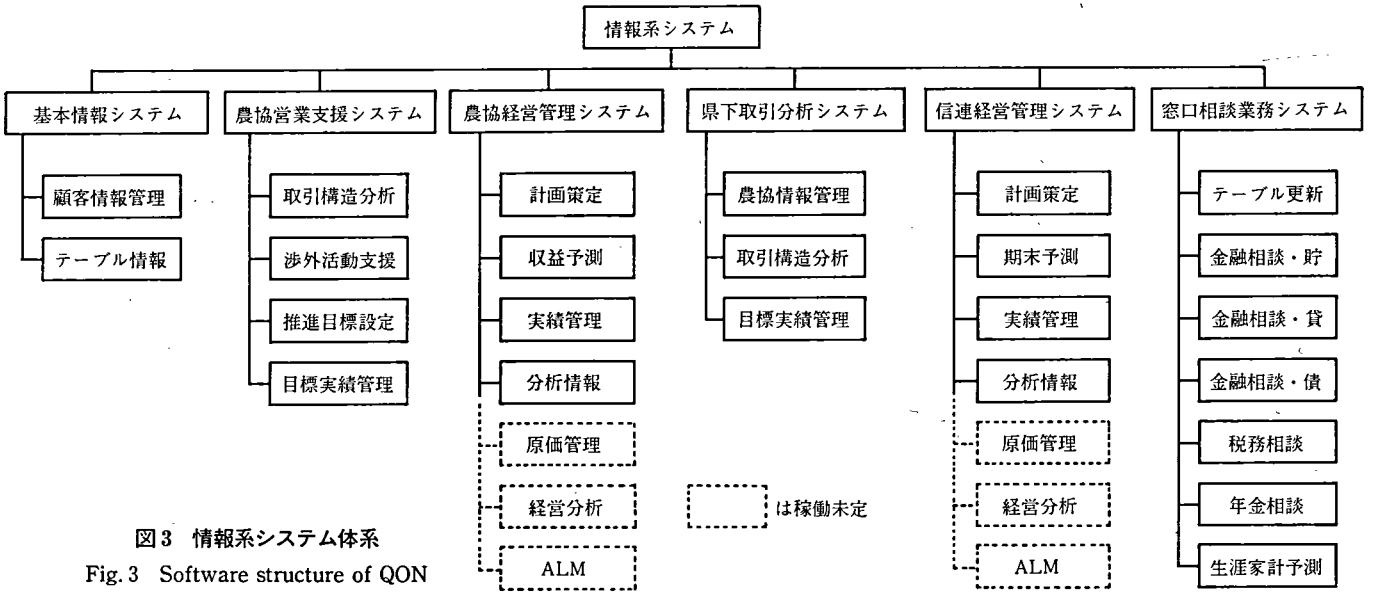


図3　情報系システム体系

Fig. 3　Software structure of QON

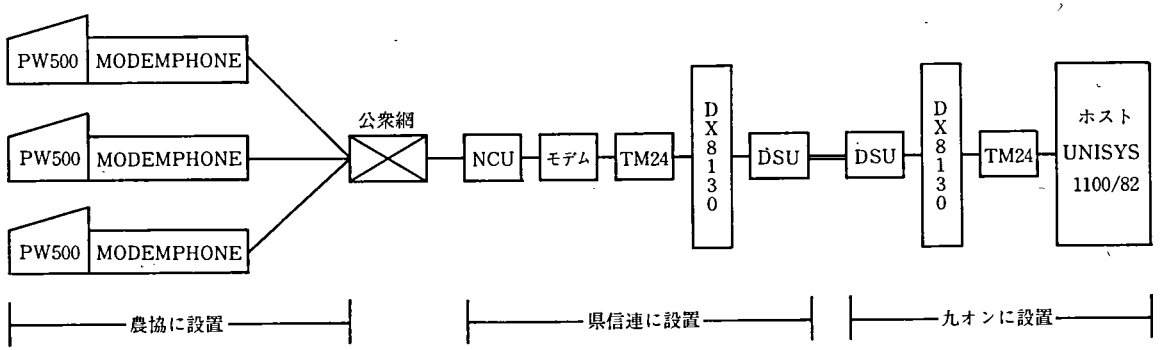


図4　情報系システム機器構成

Fig. 4　Machine configuration of QON information system

推進項目について，各種情報の管理・分析および目標・実績管理，渉外活動を支援するシステムである．

1) 取引構造分析システム……店舗または地区別の「視点」および，職業別・資格別・年令別・商品別等の「出力区分」から顧客の情報・取引状況を分析するシステムで，結果を二次加工によりグラフで見ることができる．このシステムは，効率性および成長性等を分析・把握し，渉外活動および経営管理について支援することを目的としている．

2) 渉外活動支援システム……CMF による取引先の総合情報管理をもとに，渉外活動に必要な取引先および世帯の情報を抽出するためのシステムである．このシステムは，収益性および採算性の観点から推進対象者を絞り込み，渉外活動を効率よく展開することを目的としている．

3) 推進目標設定システム……職員または農協・店舗・地区ごとに主要勘定科目・家計メイン化項目の目標を設定するシステムである．

職員目標は，毎月または年間で職員ごとの目標を設定する．設定する目標は推進目的に応じて選択できる．

農協・店舗・地区目標は毎月または年間で，農協・店舗・地区の目標を設定する．設定する目標は推進目的に応じて選択できる．

4) 目標実績管理システム……職員または農協・店舗・地区ごとに主要勘定科目・家計メイン化項目について，目標・実績値を管理するシステムである．このシステムにより，推進目標設定システムで設定した目標値に対し，実績および達成状況を月間・年間で把握することができる．

4.2.3 農協経営管理システム

農協経営管理システムは，事業運営体制の充実強化および体質強化を支援するシステムである．農協の損益項目にかかる事業計画の立案，その実績管理および実績をもとにした各種分析情報を提供するシステムであり，次のサブシステムから成る．

1) 計画策定システム……計画策定期間中の月末残高および予想される利率を入力することにより，平残・利息・利回りおよび伸び率を算出するシステムである．

2) 収益予測システム……信用事業にかかる科目について，期末における予測平残および予想される利率を入力することによって，信用事業部門の収益を予測するシステムである．

3) 実績管理システム……計画策定した項目の実績管理を行うシステムである．

4) 分析情報システム……計画策定システム，収益予測システムの利用に当たって，これらのシステムを有効に活用できるように，種々の参考情報を帳票またはグラフにより出力するシステムである．

4.2.4 県下取引分析システム

信連向けシステムで信連が県全体または信連地区レベルで県内農協を分析するシステムである．各サブシステムの考え方は農協用システムに準ずる．

4.2.5 信連経営管理システム

信連向けシステムで信連自体の経営管理システムで，考え方は農協経営管理システムに準ずる．

4.2.6 窓口相談業務システム

窓口相談業務システムは，税務相談やライフプラン等にかかる各種相談に応えるシステムであり，端末オフラインシステムである．

4.3 情報系システムの機器構成

情報系システムは早期開発が前提であったために，機器構成についても，早期稼働に支障のない効率的な構成とした（図4）．

5. 情報系システムの仕組み

5.1 システム構造

情報系システムは，九州7県のデータ量，システム変更の容易さ等を考慮し「データの管理方法」，「データの更新方法」，「データの検索方法」をポイントにしたシステム構造とした（図5）．

5.2 データの管理方法

情報管理の一元化，情報更新の容易さ，必要に応じた情報の展開およびコンピュータ資源の有効活用を考慮し，情報はデータベース化して管理している．

データベースは，利用目的に応じて日次で更新するものと，月次で更新するものがある（表1）．

5.3 データの更新方法

各種情報を展開するためのデータの収集・更新・蓄積の方法として，日次更新と月次更新の方法がある．

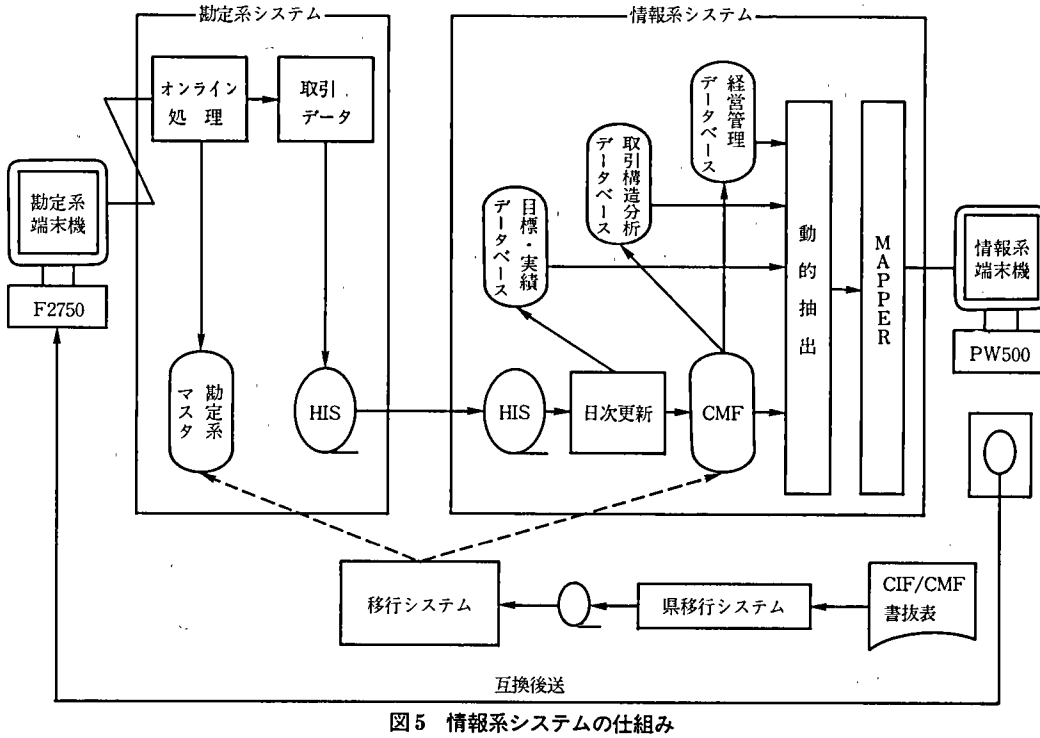


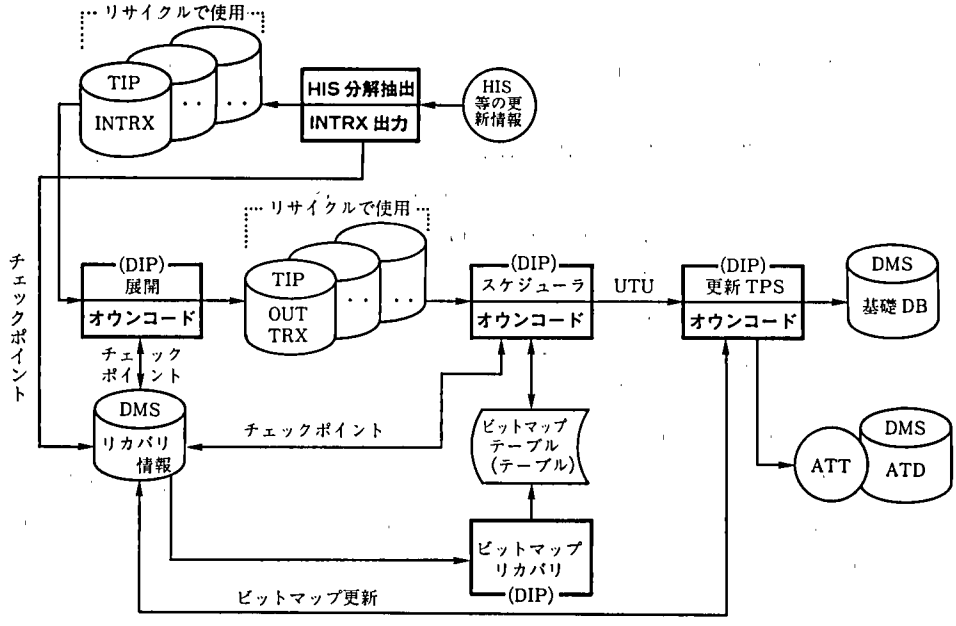
図5 情報系システムの仕組み

Fig. 5 General diagram of QON information system

表1 情報系システムのデータベース
Table 1 Database of QON information system

データベース名	管理する情報の内容	更新サイクル	使用するシステム
顧客情報管理 (CMF)	取引先の基本情報，取引状況等の情報を展開する上で必要な全情報 (取引構造分析データベースの基礎データ)	日次	顧客情報管理 渉外活動支援
取引構造分析	CMFの情報を比較的コンパクトにし，月末現在で必要な情報(前年同月・期首月等)	月次 (CMFから作成)	取引構造分析 渉外活動支援
目標	情報系端末から設置された目標値	日次	推進目標設定 目標実績管理
農協実績	月末現在での農協・店舗・地区の実績	月次	目標実績管理
職員実績	勘定系の新規取引等から職員の獲得実績	日次	目標実績管理
農協経営	主要勘定の残高・平残・利息等	月次	農協経営管理

1. システム構成図（太字はユーザ作成システム）



2. リカバリ情報

リカバリ情報は，センタダウンおよびディスク障害に対応するため，以下のような機能を持つ．

1) ビットマップテーブル復旧機能
2) HIS 入力（他システムからのデータ入力）リスタート機能
3) IN-TRX 入力（他システムからのデータ入力）リスタート機能
4) OUT-TRX 入力（他システムからのデータ入力）リスタート機能
5) TPS および各システムリスタート機能

図6 日次更新の仕組み
Fig. 6 Process of daily file updation

1) 日次更新……最新の情報を管理し展開するために，勘定系オンライン，センタカット取引で発生したデータ(HIS テープ)から毎日の CMF 等を更新する(図 6)．

① DIP の利用：日次更新システムでは，全面的に DIP を九オン向けにサマライズして採用したので，オウンコードのみの開発で済んだ．

② HIS*の利用：勘定系システムの取引パターンは数え切れないほどあり，情報系がその一つ一つに対応するためには，あまりに負荷がかかりすぎ，また整合性がとれなくなるおそれがある．このため勘定系の更新結果を HIS から抽出し書き込む方法を採っている．

2) 月次更新……データの性格上，月末現在の情報でよいものは，CMF や勘定系の各マスタや TRX からデータベースを更新する．

5.4 データの検索方法

展開する情報の内容によって，データの検索方法として動的抽出と静的抽出がある．

1) 動的抽出……情報系端末から入力された抽出条件に該当するデータを，データベースを直接検索することによって抽出する（図 7)．

① MAPPER とのインタフェース：情報系端末から入力された抽出条件を MAPPER から COBOL プログラムに渡すと同時に，COBOL プログラムを作動させる．

② データの抽出：COBOL プログラムは，MAPPER から渡された抽出条件に該当するデータを各データベースから抽出し MAPPER に渡す．

③ データの編集：MAPPER で COBOL プログラムが抽出したデータを画面出力するフォーマットに編集する．

2) 静的抽出……月次処理で，あらかじめ出力するフォーマットにデータを編集して MAPPER データベースに展開しておく．

5.5 ユーザインタフェース

情報系システムは使いやすいことが前提となるために，情報を入出力するための端末操作は，誰でも利用しやすいように次の方式を採用している（図 8，図 9)．

1) MAPPER によるユーザインタフェースの統一……情報系システムは，MAPPER, COBOL, DIP, SUFICS 等のソフトウェア，プログラム言語を使用しており，そのまま使用した場合ユーザインタフェースが異なるため，MAPPER でユーザインタフェースを開発し統一した．

2) メニュ選択……必要とする情報の選択や出力する条件の入力において，メニュ選択方式をとり，簡単に操作ができるようにしている．

3) 二次加工機能の提供……加工のための命令を知らなくても，画面に出力された情報でグラフを作成したり合計を取ったりする二次加工や画面切替等が簡単にできるように，二次加工機能を提供している．

---

* HIS：当社のオンライン・バンキング・ソフトウェア BOSS 11 のベーシックリカバリ用ロギングテープ（History Tape）で，データベース・アフタルック・レコード，トランザクション・レコード等が含まれている．

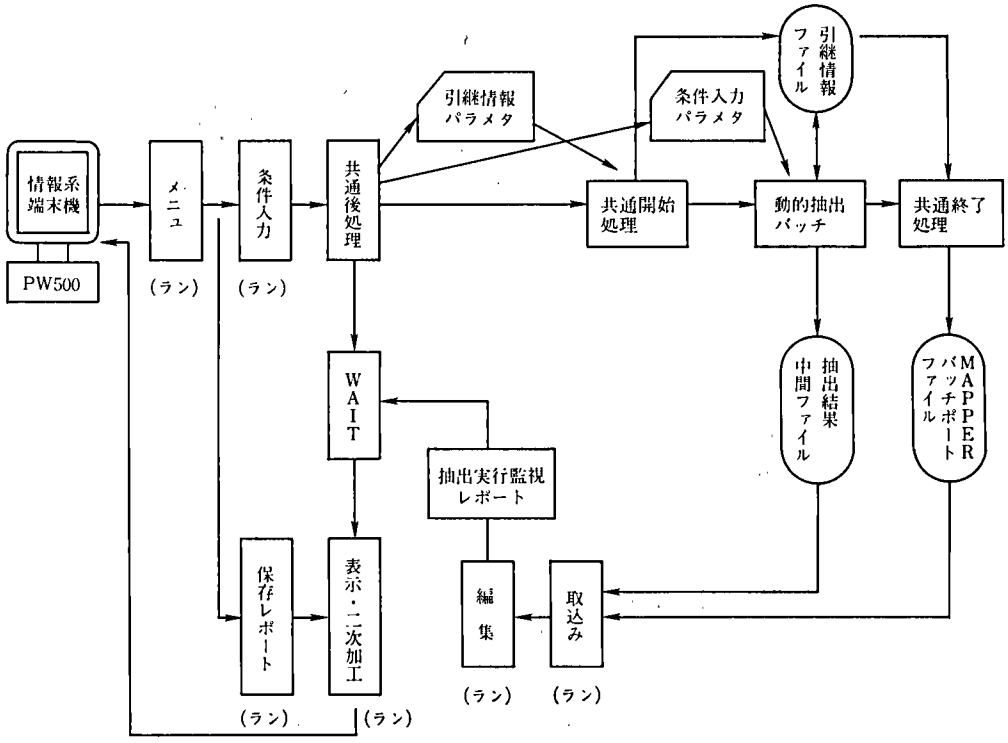


図7　動的抽出の仕組み

Fig. 7　Process of dynamic extraction

6. 開発工程

6.1 開発体制

総合系統決済システム研究会の最終レポートを受けて，昭和63年11月の第一次稼働へ向け，昭和63年4月に九オン内で次の開発体制をとった（表2）．

開発部メンバは勘定系システムの開発および運用を経験した担当者で，推進部メンバはマニアル作成等の推進部業務に詳しい担当者で構成した．また，メーカのサポートは，勘定系システムに詳しいSEと情報系システムに詳しいSEで構成した．プログラムの開発はすべて外注した．

6.2 スケジュールと進捗管理

6.2.1 開発経緯

昭和62年7月より8か月間を要して基本方針の検討・立案を行い，以降はおおむね半年サイクルで，システム設計・開発・本番を行っている．基本方針が決まっているとはいえ，具体的な仕様の決定に関しては，設計検討会の承認を得ながら設計・開発を進めるという方式のため詳細仕様の確定が遅れ，以後の工程に負荷が掛かる結果となった（表3）．

6.2.2 開発スケジュールと進捗管理

開発スケジュールの作成・進捗管理については，次のようにして行っている．

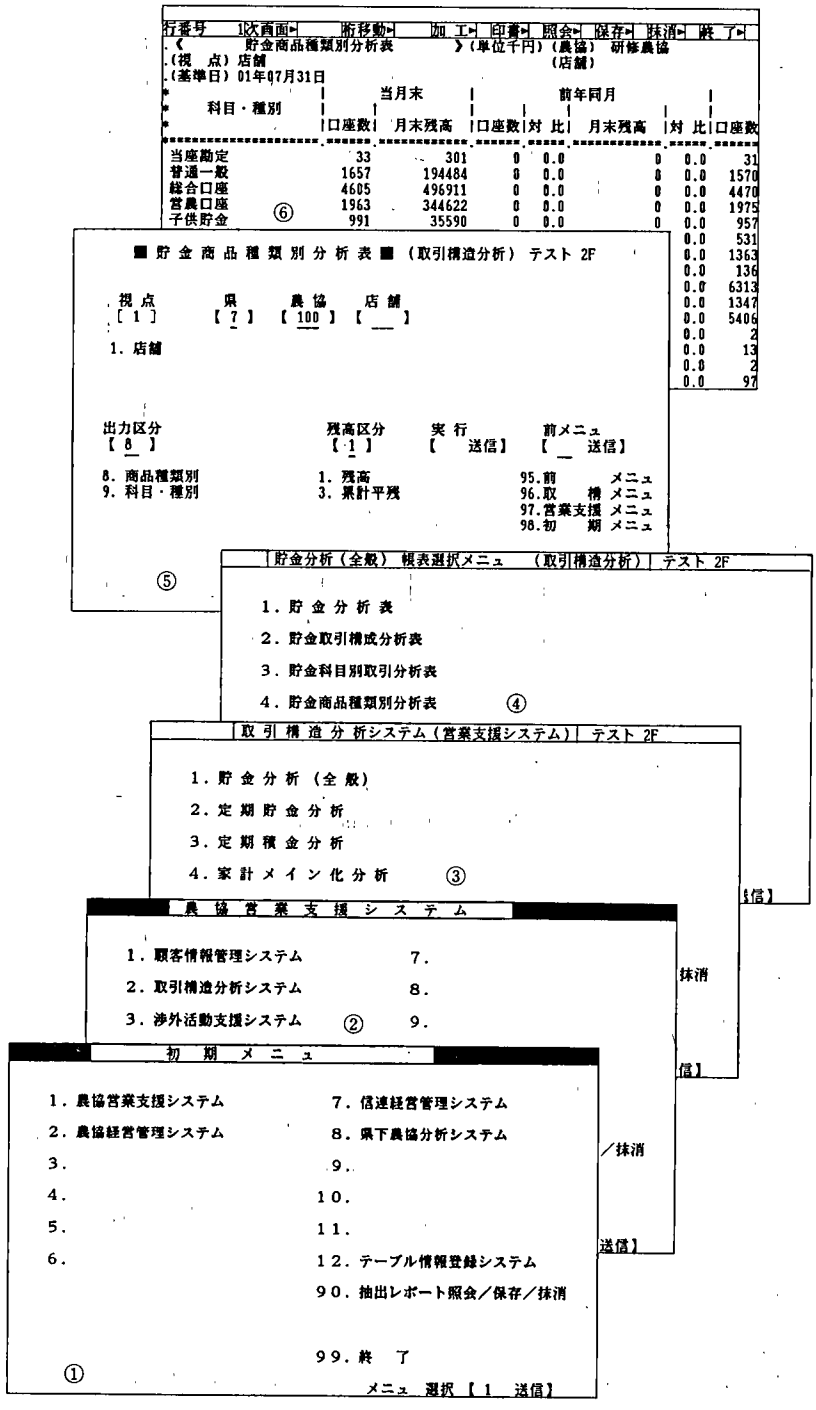


図8 実行画面例

Fig. 8 Display screen examples

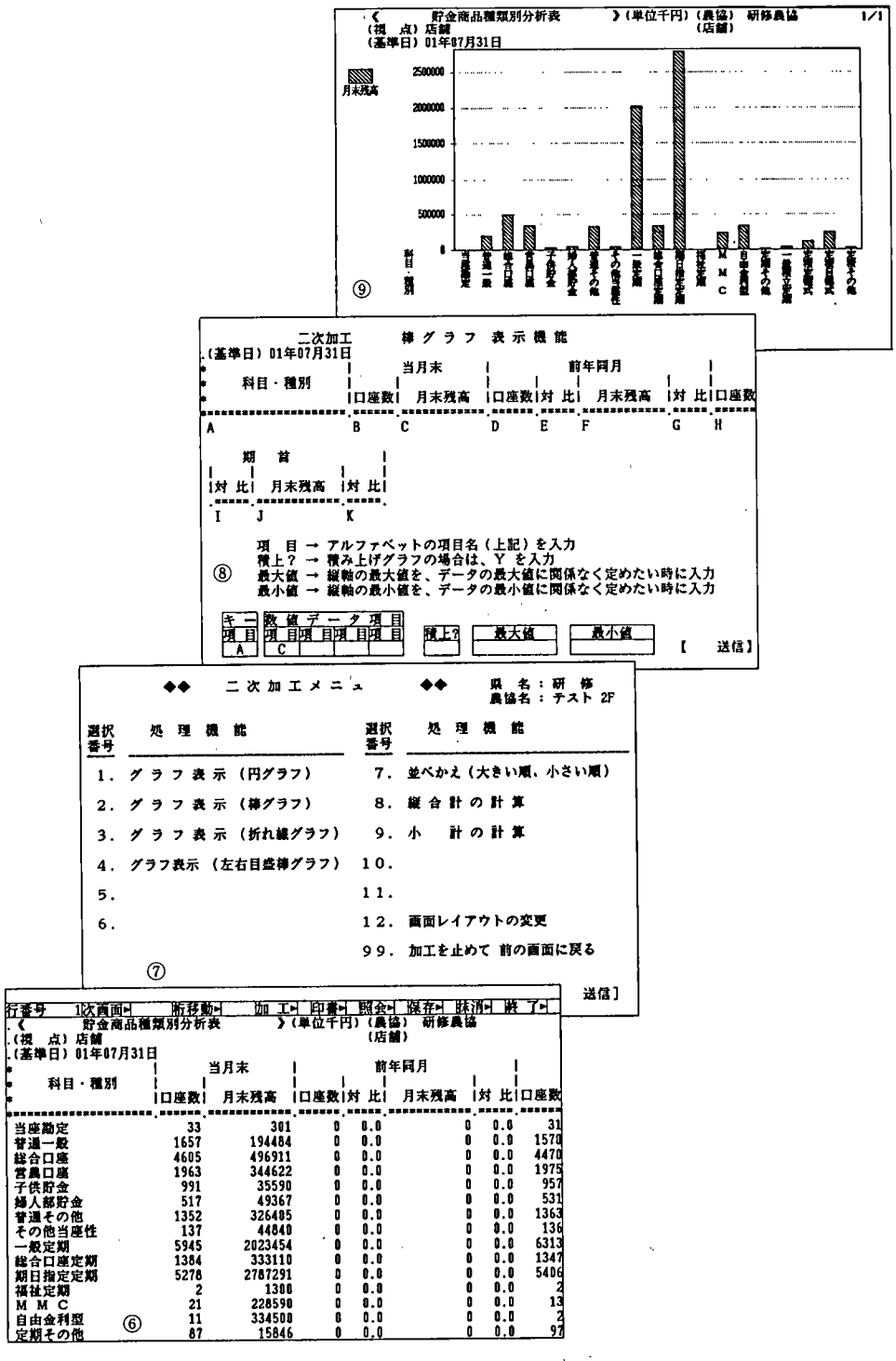


図9　二次加工実行例

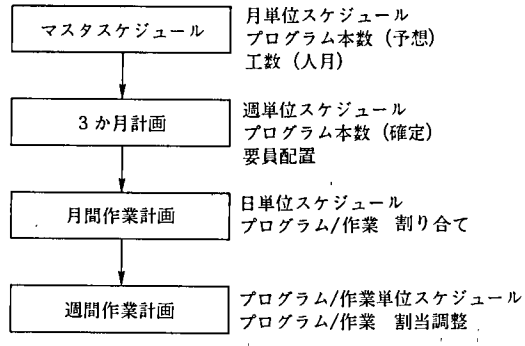
Fig. 9　Secondary manipulation examples

表2　情報系システム開発体制と作業分担(ピーク時)
Table 2　Assignment of works

	部員数	主な担当
開発部情報班	6名 (うち女性1名)	・基本設計，詳細設計，システム設計 ・プログラム仕様書，コードレビュ ・結合テスト，総合テスト ・情報系システム設計検討会事務局
推進部情報班	3名 (うち女性1名)	・端末機，回線，中継センタ機器導入関連 ・入出力画面設計 ・農協，信連のニーズとりまとめ ・研修 ・各種マニュアル作成 ・情報システム設計検討会事務局 ・情報系担当部長会議事務局
SE	4名 (うち女性1名)	・DIP導入検討と九オン版への変更（本社SEを含む） ・ユーザインタフェースの作成 ・データベース設計，動的抽出の仕組み検討 ・MAPPERラン作成
外注	10名 (うち女性4名)	・プログラム仕様書 ・プログラム作成 ・単体テスト ・MAPPERラン作成

1)　スケジュール

本システム開発においては，スケジュール作成は次に示す体系で行っている．



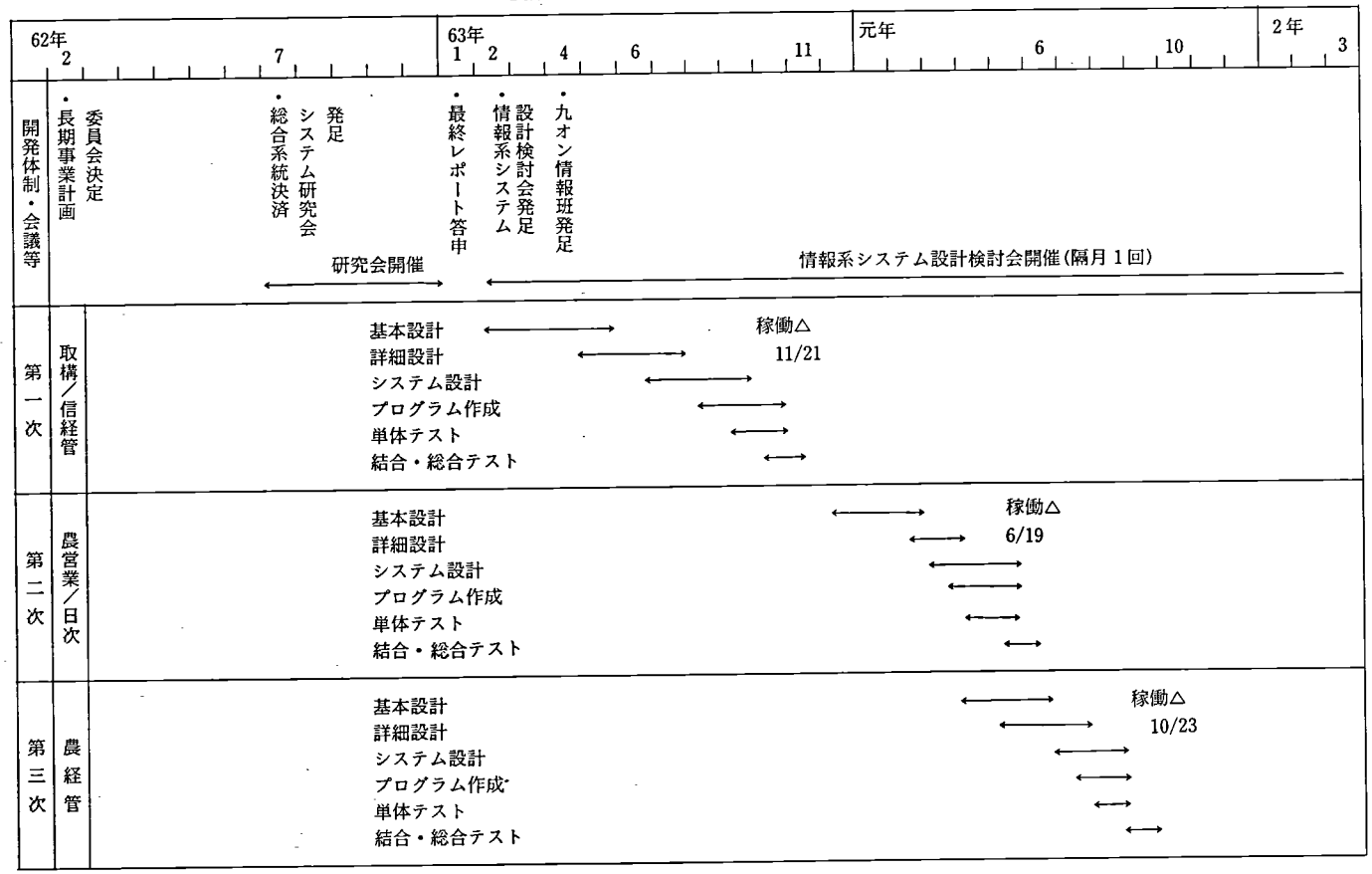
マスタスケジュールは年間計画であり，実際には6か月単位で見直しを行っている．

3か月計画は実行計画であり，サブシステムの詳細設計が終わるごとにスケジュールに反映させている．

月間作業計画は実作業に入るための作業の確認と要員の割り当てが主目的である．

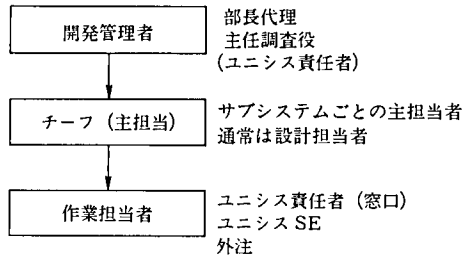
週間作業計画では，個々の作業/プログラムごとに進捗状況に応じた割り当ての調整と達成目標の確認を行う．

表3　情報系システム開発スケジュール概要
Table 3　Schedule of development



2) 進捗管理

進捗管理体制は次の通りである.



開発管理者は，主担当者のサブシステムごとの見積りに基づいて，開発スケジュールの作成・要員計画の立案・進捗管理を行う．またスケジュールの見直し・要員計画の見直し・他部門との調整・実績管理も開発管理者が行っている．

主担当者は，担当業務の設計を行い，開発量を見積り，担当業務の実質的な進捗管理を行う．また，開発環境の準備を行うとともに仕様書レビュ，コードレビュを行っている．

ユニシス責任者は，管理者の依頼に基づき要員をアサインし進捗結果を取りまとめて報告する．

進捗管理は，毎週金曜日に開発メンバ全員参加で定例のミーティングを行い，進捗の確認・スケジュールの調整・作業割り当て変更を行っている．

6.3 開 発 実 績

昭和 63 年 4 月より，14 か月間の開発実績は次の通りである.

	本 数	ステップ数	実績工数(人月)	見積工数(人月)
COBOL	556	350,411	244	350
MAPPER	621	85,856	52	58
計			296	408

詳細は表 4 に示す．このような大規模システムを短期間に開発できた要因としては，次の事柄が挙げられる.

1) KINGS の導入……KINGS の導入により，情報系システムの要件/機能，入出力設計，データベース設計，適用ノウハウ等を最初から入手できたため，開発の企画・設計の段階で少なくとも 6 か月間は開発期間を短縮できたと考えられる.

2) DIP の導入……DIP の導入により，情報系仕組みソフトウェアの設計開発作業が一部カスタマイズ部分を除いて不要となった.

今回の開発における DIP 導入の効果は，

① 抽出・加工/蓄積・更新 設計 16 人月

② 抽出・加工/蓄積・更新 開発 110 人月

の開発工数の削減と開発期間短縮と考えられる.

3) 検証の自動化……情報系マスタの日次更新の更新結果は，勘定系マスタとの照合により検証が可能であることに着目し，全科目の検証プログラムを開発した.

表4 開 発 実 績
Table 4 Step of program

システム名	COBOL		MAPPER	
	本 数	ステップ数	本 数	ステップ数
顧客情報管理	12	10,346	58	5,012
取引構造分析	112	69,659	74	7,199
渉外活動支援	11	6,715	54	4,667
推進目標設定	27	44,901	36	3,066
目標実績管理			73	6,151
信連経営管理	45	20,038	193	43,190
共 通	53	13,655	34	5,013
日次更新	177	105,905	20	2,989
テーブル情報			10	3,952
合併分割	84	42,222		
運用管理			69	4,617
検 証	35	17,289		
計	556	330,370	621	85,856
登 録 集	327	19,711		
合 計		360,441		

もしも検証システムがないとすると，検証を行うためには入力 HIS テープのダンプと出力データベースのダンプを取り，照合する必要がある．それぞれに専用のダンプ・ツールを開発するか，更新 TPS すべてに入出力スナップ・ダンプ・ルーチンを組み込む必要がある．さらに一取引ごとのダンプをもとに手計算で結果の検証を行う必要があるため，本番データを用いた総合テストの，1万件×10日というボリュームでは実際上，不可能に近い．

総合テストでの検証システムの使用結果を見ると，検証システムでエラーを検出したデータの追跡・調査でも，2週間掛かっている．もし検証システムなしに手作業で行った場合には，検証要員として5名を投入したとしても，少なくとも3か月は掛かったものと推定される．

4) MAPPER ラン　チェックポイント・ログ機能……MAPPER ランは，インタプリタという特性から単一のランのデバッグは極めて容易である．

しかしながら，情報系では複雑な仕組みで動いているために，どこで異常終了したかがわからない．そこで現象再現からスタートして原因追求を行うことになり，通常のランでは数分で原因がわかるものが，数時間あるいは1日かかる場合がある．

この対策として，すべてのランで定義しているエラー処理ルーチンでチェックポイント情報をロギングするようにした．

またバッチポート機能で外部のプログラムを起動する場合，指定したパラメタ，JCL（ジョブ制御言語）等をロギングし，原因の追求や現象の再現を容易にした．

効果を数字で表すことは困難であるが，MAPPER ラン開発において非常に有効であった．

本番後もエンド・ユーザからのトラブル通報時に，センタ側でログを解析することにより，迅速かつ正確に原因究明を行えるため連絡を受けて短時間のうちに

対応できている（通常 30 分以内）.

5) プログラムのパターン化……情報系においては，同一の仕組み上で動くようになっているので容易にプログラムをパターン化できる．この場合，類似プログラムを複写して相違部分のみ変更することによりプログラムを作成するため能率が良い．

COBOL の約 60 %がこれに該当し，この部分の開発は実績によれば通常の半分程度で済んでいる．

6.4 開発上の留意点

システム開発を短期間で，効率的な信頼性の高いシステムを構築するために，次の点に留意し，開発を進めた．

6.4.1 システム開発工程基準の遵守

情報系システムは，62 年度に見直し制定した「システム開発工程」，「システム開発手続」を遵守したシステム開発を行った（図 10）.

6.4.2 プログラムの品質管理

プログラム仕様書，プログラム作成の外注および随時変更に対応しやすいように単純で明快なプログラム作成をめざし次の方法を採った．

1) ジャクソン法（JSP）に基づき，プログラム構造の統一，モジュール化の徹底を図った．

2) プログラム仕様書とプログラム作成をそれぞれ担当分けして相互チェックを行い，また交互にプログラム仕様書とプログラム作成を担当することにより，プログラム技術の向上をめざした．

3) 全プログラムに対してプログラム仕様書のレビュ，コードレビュを行い，プログラムの質の向上と開発効率を良くした．

4) 外注のメンバに対して，開発に着手する前に開発工程・手続の研修を行い，レビュに全員参加することによって，システムの理解・標準化等の徹底を図った．

6.4.3 テストの実施と検証

情報系データは勘定系データを基本としており，各テストでは勘定系の結果と同じ内容になることを前提に，テストの実施・検証を行った．

1) CMF の日次更新に関するテストは，すべての勘定系取引（オンライン，センタカット他）のデータを作成することは不可能であるために，ある一定期間毎日，勘定系本番 HIS を使用して仮本番運用を行うことによって実施した．

テスト結果は，検証システムを構築し更新後の CMF 内容と勘定系各マスタを突き合わせ検証した．

2) 月次処理に関するテストは，勘定系で出力する還元帳票と照合し検証した．

3) 移行処理に関するテストは，各県より移行データを作成してもらい，仮移行処理を行い検証した．

4) MAPPER ランについては，研修用に展開したデータベースを使用して内部で情報系端末より各情報の入出力を行うとともに，稼働前に農協・信連の担当者に対して研修を実施し，意見を聞くことによってシステムの検証とシステムの改善をすることができた．

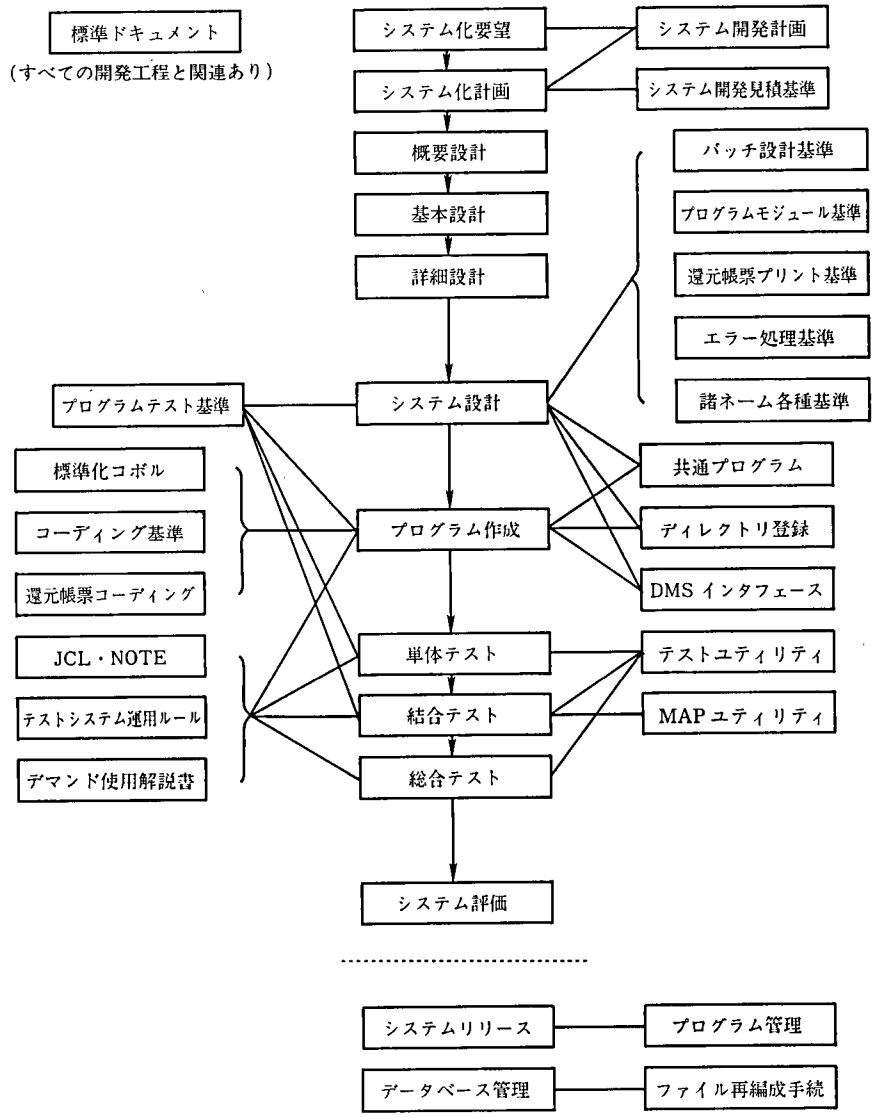


図10　システム開発工程

Fig. 10　Process of system development

6.4.4　内部承認制度による品質向上

開発各工程において次の内部承認制度にその都度はかることにより，システム全体の品質向上，運用効率の向上をめざした．

1)　リリース委員会……システム開発の工程の節目において，システムの品質を審査することによって，システムの信頼性・安定性・効率性を高めることを目的とする．開発部および運用部の各担当代表者で構成する．

2)　開発調整会議……開発部および運用部におけるシステム開発，および運用に関わる項目の協議を行う．各部調査役および各担当チーフで構成する．

3)　プロセス委員会……バッチ処理システムの体系を検討することによって，本番

稼働システム運用の安定性・効率性を高めることを目的とする．運用部の各担当代表者で構成する．

7. おわりに

平成元年10月の農協経営管理システムの稼働をもって，当初予定した九オン情報系システムの基本的構築を終了した．この情報系システムの開発は，短期間の開発，システム開発工程の手続にそった開発，プログラム開発の全面的外注等，九オンでも初めての方法による開発であった．開発方法の一つとして，今後開発するシステムの指針になり得ると思う．

しかし，今回はすべて新規の開発であったことが良い結果につながったと思われる．今後のシステム変更・追加等にいかに対応するかがわれわれの課題である．

またシステムの利用の面で，農協の参加および利用をいかに定着させるかが，信連と九オンが一体となって今後取り組まなければならない課題である．

執筆者紹介 前 山 泰 治 (Taiji Maeyama)
昭和28年生．51年同志社大学法学部法律学科卒業，同年佐賀県信用農業連合会入会，53年九州地区農協オンラインセンタ出向．現在，開発部に所属．

平 嶋 孝 正 (Takamasa Hirashima)
昭和22年生．44年九州大学工業教員養成所工業化学科卒業．45年日本ユニシス(株)入社，製造系ユーザのSEサービスを担当．60年より九州地区農協オンラインセンタにて情報系を担当．現在，九州支店に所属．

オフコンにおける大規模システム開発の事例

An Example of Large-scale Systems Development Based on Small Business Computers

市 川 常 男

要 約 近年オフィス・コンピュータ（以下オフコン）の使われ方は，経理／給与に代表される業務処理を行う単一オフコン処理から，オンライン・リアルタイム処理のホスト機に使われるまで，その使用形態が多様化してきている．

したがってシステム開発の方法も，その使用形態および規模によって異なってくる．

本稿では，オフコンによる大規模リアルタイム・システムを構築しているA社の勘定系システムの開発過程を述べることにより，オフコンにおける大規模システム開発の一方法論としたい．

記述は，システム開発工程にそって以下の順で述べる．

1) システムの概要
2) 開発見積りとスケジューリング
3) 開発体制と役割分担
4) 開発支援ソフトウェア
5) 開発進捗管理
6) システム・テスト
7) 開発プロジェクトの評価

Abstract Recent uses of small business computers have become more and more diversified, ranging from single-purpose data processing for such applications as accounting and payrolls to their use as central host computers for online real-time systems. Then, the ways to develop systems based on small business computers are necessarily various according to the different environment where they are used and the scale on which systems are built.

This paper describes the process by which an accounting system has practically been developed at A company where a large-scale real-time system built upon small business computers is now in operation.

Details are given in the following order in parallel with the systems development paocasses:

1) scope of the system
2) installation estimates and scheduling
3) development organization and job assignment
4) development support software
5) development process management
6) system testing
7) evaluation of the project

1. はじめに

A社では，金融業務の勘定系システムをオフコンを使った分散処理で運用してきた．

このシステムは，急成長下の支店窓口業務をスムーズに処理できたこと等，システム化の効果は大きかった．

しかし，このシステムも機械の老朽化，運用負荷の増大，システム拡張が困難等，種々の問題が発生してくるようになった．これらの問題を解決することと，支店の営業および管理データの即時提供，自動機器の監視機能の確立，24時間，365日稼働システムの構築を目的に，新勘定系システムの開発がスタートした．

この開発は，システム企画書の作成から本番までが1年という短い工期の開発となり，非常に困難な開発であったが，客先・当社・協力会社の三身一体の開発プロジェクトにより，予定通りの本番稼働が実現した．

本稿では，今回の新勘定系システムの開発プロジェクトを，システム開発工程にそって振り返ることにより，オフコンにおける大規模オンライン・システム開発の一方法論としたい．

2. システムの概要

A社のシステムは，大きくは勘定系システムと情報系システムに分かれるが，本稿では勘定系システムについて述べる．

勘定系システムは，顧客データベースを持つ中核システムとなる地区センタ・システム，これら地区センタを本社でコントロールする運用センタ・システム，支店の窓口業務を行う支店システムの3システムから成る．

地区センタ・システムには，大型オフコンのUNISYSシリーズ8/モデル8000を複数台ずつ7か所に分散して設置し，各地区のオフコン間をNTT(日本電信電話株式会社)のDDX-P網で結んでいる．

運用センタ・システムには，地区センタと同様にUNISYSシリーズ8/モデル8000を設置し，自動運転を行っている各地区センタのオフコンをNTTのDDX-P網および電話網で遠隔制御している．

また支店システムには，小型オフコンのUNISYSシリーズ8/モデル100 IIや自動入出金機を設置し，専用回線で地区センタと接続し，支店の業務処理を行っている．

このように，オフコンを水平・垂直に分散設置し，用途に合ったネットワーク網で結ぶことにより，コスト・パフォーマンスが高く，安全性の高いシステムを実現している．

図1にシステムの概念図を示す．

2.1 二重系システム

地区センタ・システムは，自動入出金機(ATM)のサービスのため，24時間稼働でリアルタイム処理を行っている．このため本システムでは，各種の障害対策がとられている．

この障害対策の大きな柱となっているのが，ディスクの二重化とCPUの二重化である．顧客マスタ等の重要ファイルは，2台のディスクに，メインファイルおよびサブファイルとして二重に持たれている．業務処理は，入力時にメインファイルからリードし，更新時には，メインおよびサブの両ファイルを更新する．

ディスクでハードウェア障害が発生すると，誤り状況が運用センタ・システムの地

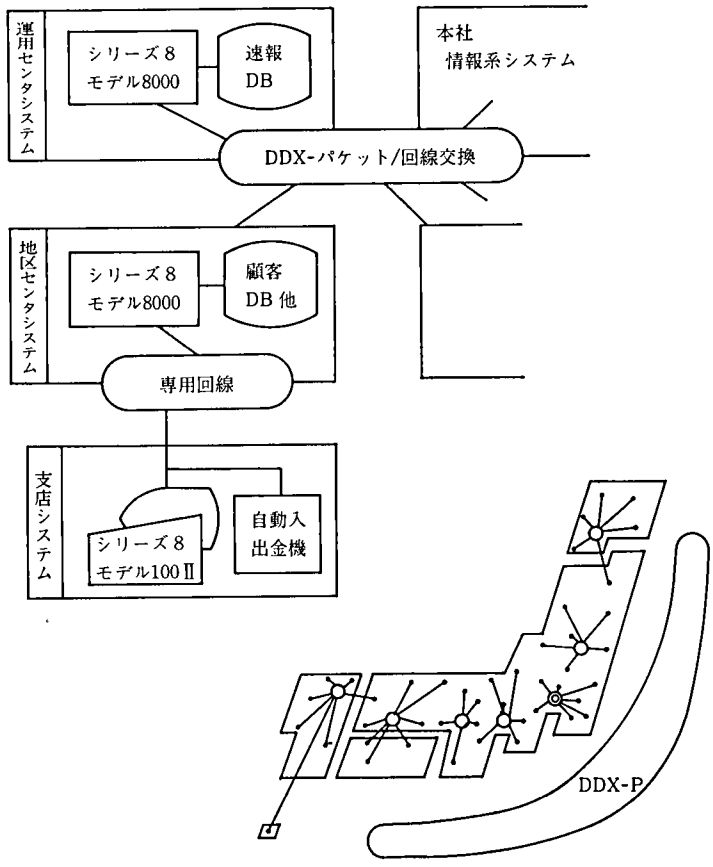


図1　システム概念図
Fig. 1　Outline of system

区センタ監視画面にブザーと共に表示される．

運用管理者は，誤り状況を判断し地区センタの遠隔制御機能を使って障害ディスクをはずし，縮退システムとして再立ち上げを行う．これらの機能は，すべてユーザ・アプリケーションとして開発されており，オフコンとしては最新のシステムになっていると思われる．

次にCPUの二重化であるが，この機能は地区センタには2台のオフコンをペアで設置し，通常は各々のオフコンで別々の支店群の端末処理を行っている．

一方のCPUで障害が発生すると，SC（システム監視用パソコン）から，運用センタの監視画面に障害が表示される．これを受けて運用センタでは，障害の発生したCPUとペアになっているCPUの再立ち上げを行う．この操作によって，今まで二つのCPUに別々に接続されていた支店群の処理が正常な1台のCPUで処理できるようになる．

CPU障害発生時のリカバリ手順を示すと図2のようになる．

図中のリカバリ手順③のディスク切り替えでは，デュアルポート機構(シリーズ8/モデル8000に装備されているディスク切替機能)を使って，ディスクの接続を障害の

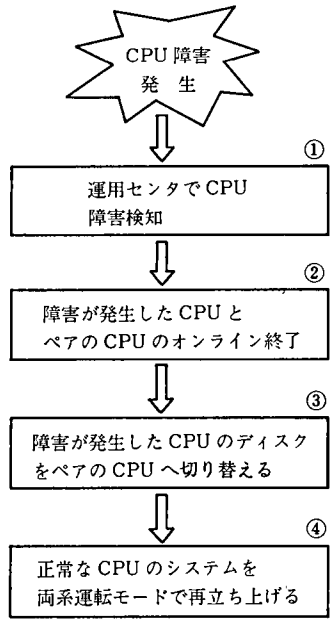


図 2 CPU 障害のリカバリ手順

Fig. 2 Recovery process for processor trouble

発生した CPU から正常に稼働している CPU に切り替える.

手順の中に回線の切り替えがないのは，回線切り替え装置を使わず直流分岐装置を使って，支店の回線を両 CPU に接続しているためである．電文の選択は回線のオープンの仕方を立ち上げ時のコマンドを変えることによって行っている.

A 社勘定系システムは二重系システムを採用することにより，オフコンのシステムではあるが，汎用機並みの障害対策を実現しており，障害発生時の回復時間は，ディスク障害で 10 分，CPU 障害で 20 分程度となっている.

図 3 に二重系システムの構成図を示した.

2.2 業務プログラム(AP)

地区センタ・システムと運用センタ・システムは，OS として DPS10 を使い，この OS の下で稼働する TPS（トランザクション処理システム）の下で業務プログラムを稼働させ，業務処理を行っている(図 4).

また，支店システムでは OS として DPS IVを使い，今回のシステムのために開発した LCP（ライン・コントロール・プログラム）の下で業務プログラムを稼働させ業務処理を行っている(図 5).

3. 開発見積りとスケジューリング

今回のシステム開発は，旧システムの分散処理方式を継承し，業務処理の中核部分の仕様は現行の仕様とする等，コンバージョンの比重が高いシステム開発であったものの，回線制御プログラムを従来のローカル AP から，TP(トランザクション処理プログラム）に変更したためのプログラムの改造，二重系システムへの考慮等のため，

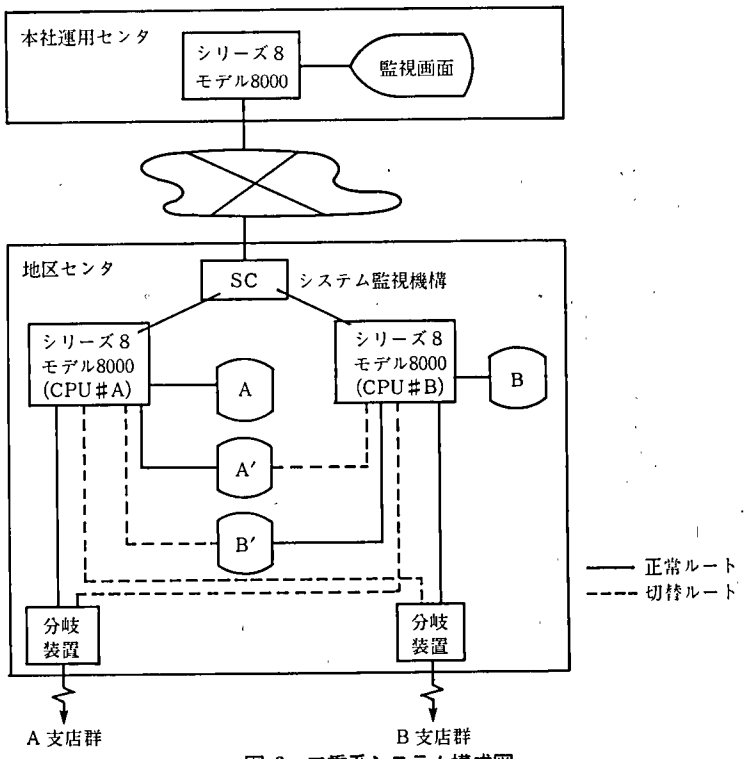


図 3　二重系システム構成図

Fig. 3　Configuration of duplex system

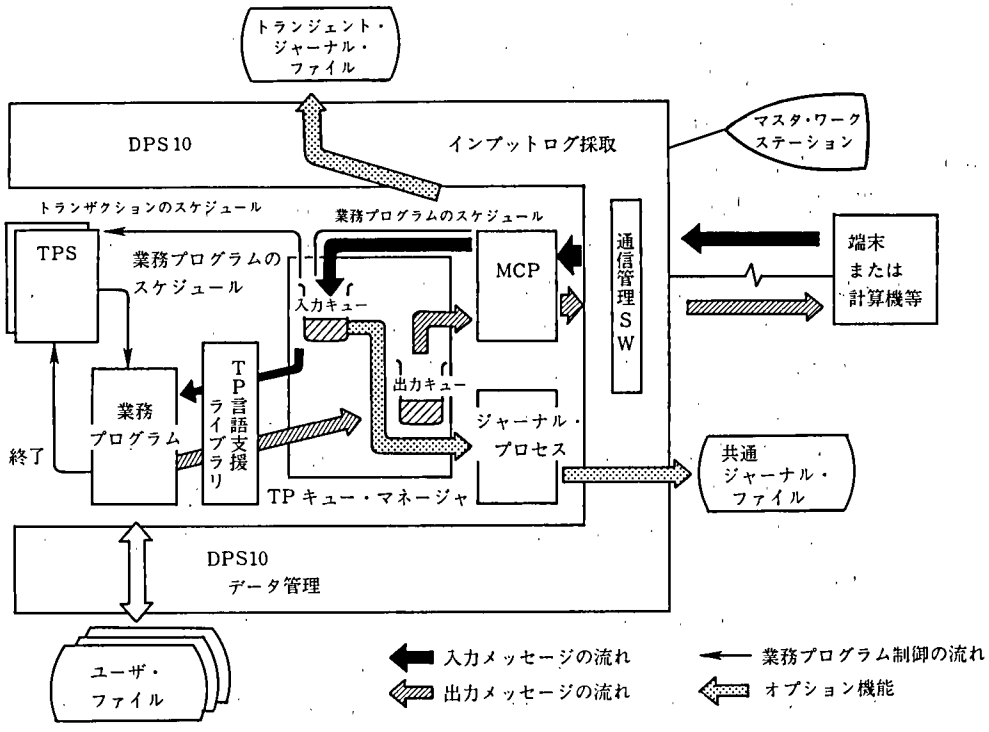


図 4　DPS10TP システム構成図

Fig. 4　Structure of DPS10TP systems

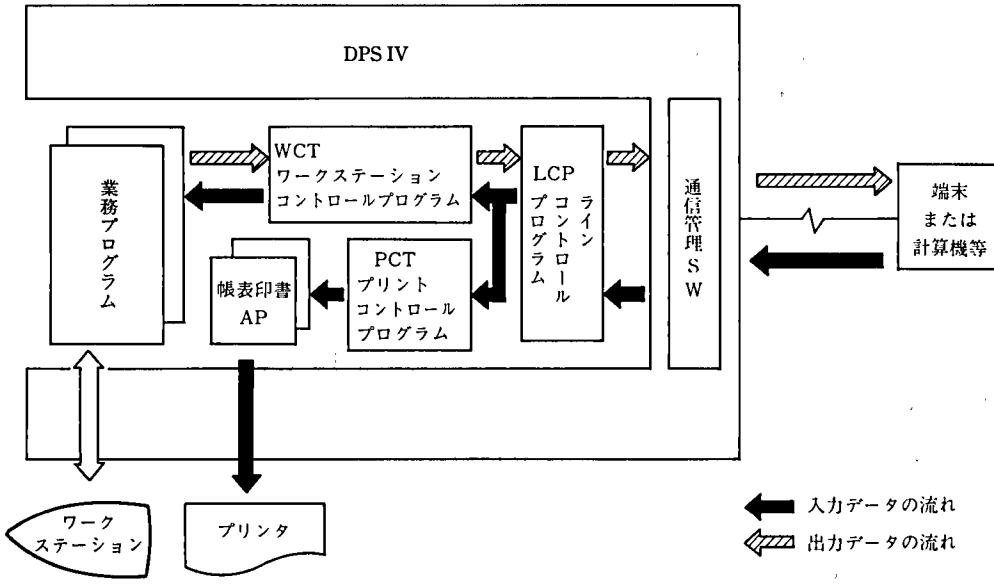


図 5　DPS IV-LCP システム構成図

Fig. 5　Structure of DPS IV-LCP systems

新規のシステム開発並みの見積りが必要となった．

また，開発言語についても COBOL とプログレス -II（シリーズ 8 用コンパイラ）の二つの言語での開発になる等，見積りのむずかしさがあった．

二つの言語の使い分けについては，処理スピードの要求される地区センタでのリアル更新処理は COBOL を支店システムの画面入出力，および帳表の出力は生産性の高いプログレス-II（P-II）を採用した．

3.1 開発工数見積り

開発工数の見積り方法もいろいろあるが，今回のプロジェクトでは，各システムの規模をプログラム本数およびステップ数で見積り，これに各言語別の生産性基準値を利用して総工数を算出した．この見積り方法を段階的に見てみると次のようになる．

1)　プログラム本数，ステップ数の算出……地区センタ／運用センタ／支店システムに分割し，次に業務ごとにブロック化する等，トップダウンで構造を作り，プログラム本数を見積った．ステップ数については，現行のプログラムのステップの 1.5 倍とする等の係数を掛けて算出した(表 1)．

2)　生産性見積り基準表の作成……システム別，新規／移行別，オンライン／バッチ別に難易度，言語別の生産性見積り基準表を作った．言語別の生産性基準値は，COBOL は汎用機で使われている生産性基準を使い，プログレス-II については 1 本当たりの生産性基準を使い，A 社での過去の実積を加味した係数を基に算出した(表 2)．なお 1 人月は 150 時間としている．

3)　システムごとの開発工数見積り……1)で算出したシステムごとのステップ数を，2)で算出した生産性基準値で割って，総工数を見積った．

この見積りの結果，今回のシステム開発の総工数は，約 300 人（約 347,000 ステッ

表 1　開発規模見積り表

Table 1　Estimated table of development scale

区分				見積りプログラム本数 [本]	見積りステップ数 [ステップ]
地区センタ	新規	オンライン	COBOL	65	86,500
		バッチ	COBOL	20	10,000
			P-Ⅱ	50	25,000
	移行	オンライン	COBOL	75	136,000
運用	新規	オンライン	COBOL	20	10,000
					25,000

表 2　開発生産性見積り基準表

Table 2　Standard table of estimation in development productivity

区分					COBOL 生産性 [ステップ/月]	P-Ⅱ 生産性 [ステップ/月]
地区センタ	新規作成	オンライン	難易度	大	600	1000
				中	800	1200
				小	1000	1450
		バッチ	難易度	大	800	1200
				中	1000	1500
				小	1200	1800
	移行	オンライン	難易度	大	900	1500
				中	1200	1800
				小	1500	2200
		バッチ	難易度	大	1400	1800
				中	1500	2300
				小	1800	2700
	新規	オンラ	難			1000

プ) となり，オフコンとしては大規模なオンライン・システムの開発となった．

3.2　開発作業ブロック図

開発をスタートするに当たり，開発期間が 1 か月，工数 10 人月のブロックを考え，それを開発工程ごとに積み重ねた開発作業ブロック図を作成し，開発作業を視覚的にとらえてみた．

まず開発工程を表 3 に示す標準工程とし，それぞれの工程ごとに，工数割合および時間割合を開発内容を考慮して設定する(表 3)．

次に，各工程ごとにブロックを期間割合で横に並べ，縦にはシステムごとの総工数を10人月で割った数のブロックを積み重ねる．積み重ねたブロックの下には，各開発工程で行う作業項目を列記し，ブロック内での開発作業の目安とした．このようにしてできた開発作業ブロック図が，次の開発工程表作成の基礎になる．

図6に地区センタ・システムの開発作業ブロック図を示した．

表3　開発工程・工数・期間割合表

Table 3　Proportion table of cost and term in development process

工　程　名	工数割合 (%)	期間割合 (%)
基　本　設　計	10	20
詳　細　設　計	25	20
プログラム作成	45	30
システムテスト	15	20
受け入れテスト	5	10
合　　計	100	100

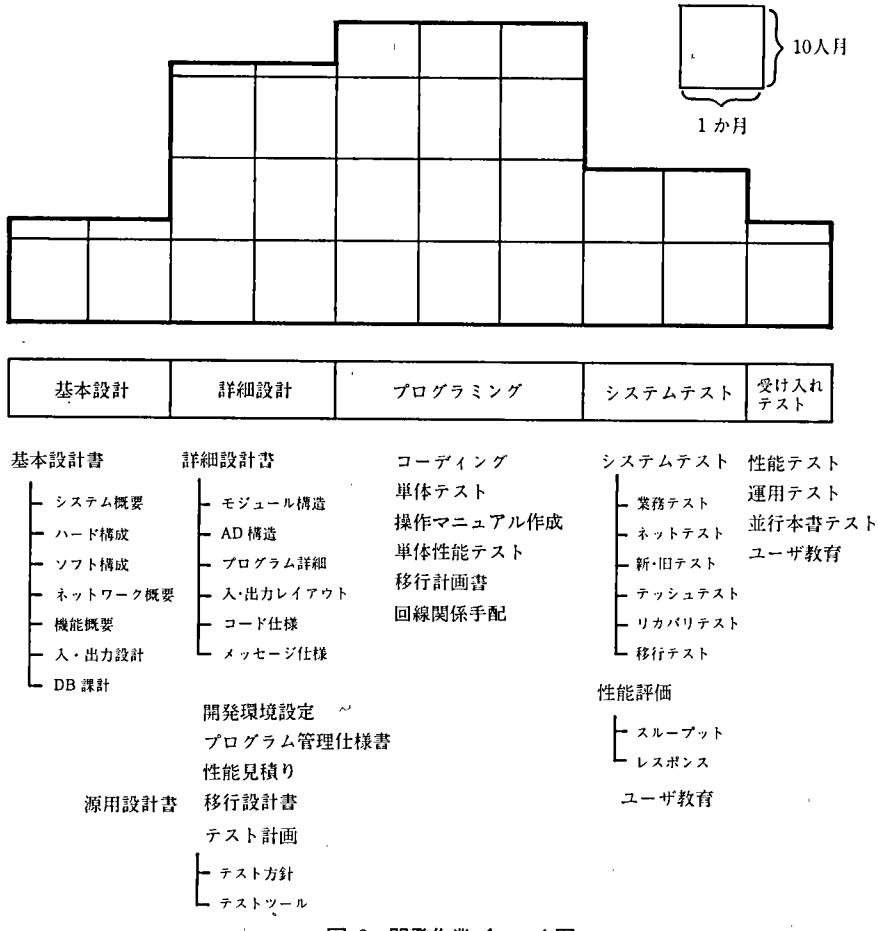


図6　開発作業ブロック図

Fig. 6　Block chart of development

3.3 開発工程表の作成

工程表の作成には，ガントチャート，バーチャート，PERT 等の方法があるが，各作業の順序と継がりが表示できること，クリティカル・パスが見出せる等の長所を持った PERT のアロー型ネットワーク工程表を使った．

工程表の作成は以下の手順で行った．

1) 各システムごとの開発作業ブロック図を使って，各作業ごとのアローの長さ（工程表の矢印の長さで，工期の長さを示している）とアローの下に記入する工数を見積る．

2) 各工程ごとに，作業を入れたネットワーク工程表を記入し各システムごとの関連性および，クリティカル・パスをチェックする．

3) ネットワーク工程表の上部にイベント欄を設けて，各工程ごとのチェックポイント日付とマシンの搬入等のイベントを書き込む．

例として，地区センタ・システムの工程表を図 7 に示した．

4. 開発体制と役割分担

1 年間で開発を仕上げるための体制として，旧システムを熟知している当社の SE を中核とし，客先，協力会社の三者で開発プロジェクトを作った．

さらにプロジェクト・リーダとして，客先，当社から 1 名づつ選出し，各システムごとには，客先，当社，協力会社からサブリーダを選出し，合計 11 名からなるリーダ会議を設けた．

このリーダ会議は，基本設計工程では 2 週間に 1 回，詳細設計以降は 1 週間に 1 回の割合で，進捗状況のチェックを行った．

約 1 年間での開発は，客先，当社，協力会社の好リレーションがなくては成し得なかったことであり，1 年後に本番が迎えられたことは，この協力体制のおかげであったと思う．

三者の役割分担を開発工程ごとに整理すると，表 4 のようになる．

5. 開発支援ソフトウェア

今回の開発では，支店システムの画面プログラムについてはシステムの品質向上および生産性向上を目指して，シリーズ 8 に準備されている PRODUCE（開発支援ソフトウェア）を採用した．

このことは開発での生産性向上だけでなく，本番後のメンテナンス時に効力を発揮することになった．

今回使用した PRODUCE の機能は，次のようなものである．

1) 画面，帳表等の仕様書のコンピュータ出力

2) プログラムの原始コードの自動生成

3) 画面，帳表プログラムの修正をイメージ修正で行える

4) 原始コードと一致したドキュメント類の出力

参考として，今回 PRODUCE を使って開発した画面プログラムの画面レイアウトとファイル・レイアウトを図 8，図 9 に示した．

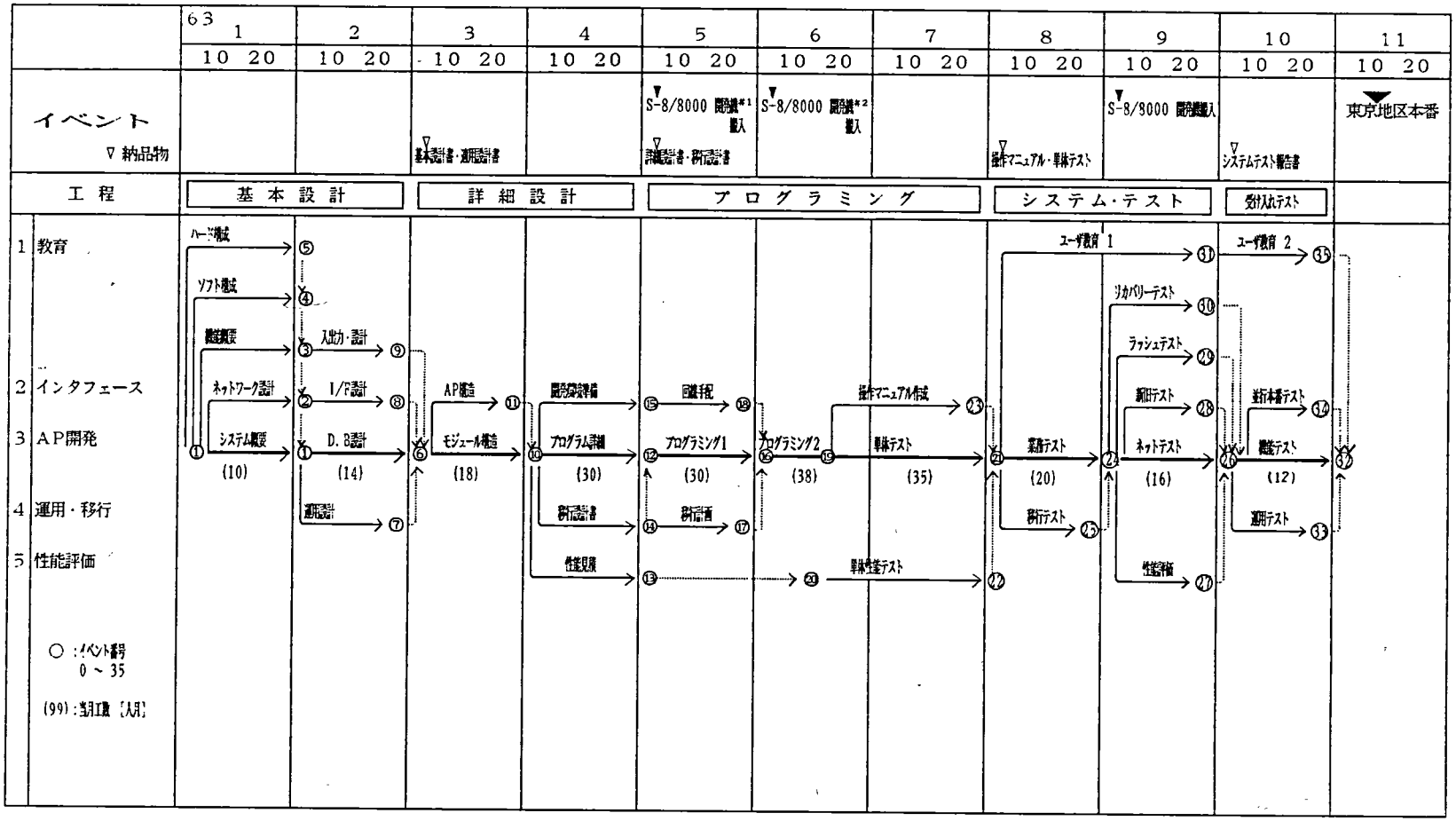


図 7 地区センタシステム工程表

Fig. 7 Scheduling chart of division system

表4　開発時の役割分担
Table 4　Assignment of development

○：メイン担当，△：サブ担当

工　程	作業内容	客先	当社	協力会社
要求定義	システムの目的の明確化	○	△	
	システムの目標と機能の明確化	○	△	
	システム開発方針の明確化	○	△	
	システム開発規模，HW，SWの決定	△	○	
	開発／移行スケジュールの決定	○	○	
	開発基準書の作成	○	○	
基本設計	基本設計書の作成	△	○	△
	運用設計書の作成	○	○	
詳細設計	詳細設計書の作成	△	△	○
	開発環境の設定	△	○	
	プログラム管理仕様書の作成	△	○	△
	性能見積り		○	
	移行設計書の作成	△	○	○
	テスト仕様書の作成	△	○	△
プログラミング	コーディング作業			○
	単体テスト		△	○
	操作マニュアルの作成	○	△	
	単体性能テスト		○	○
	移行計画書の作成	○	△	
	回線関係の手配	○	△	
システムテスト	システムテストの実施		○	○
	システムテストの評価	△	○	△
	性能測定と評価	△	○	
	運用マニュアルの作成	○	△	
	エンド・ユーザ教育	○	△	
受け入れテスト	機能テスト	○	△	
	運用テスト	○	△	
	並行本番テスト	○	△	
	移行テスト	○	△	
	エンド・ユーザ教育	○	△	

6. 開発進捗管理

開発の進捗を正しく認知し，スケジュールとの差を正しくつかみ，ずれを補正する手段を取ることは，大規模システム開発を成功させる上で，最も重要なことである．

今回のプロジェクトでは，このために毎週1回，客先，当社，協力会社のリーダが集まっての進捗会議を行った．この会議では，進捗状況を%表示した円を使った開発フェーズ進捗状況表，およびプログラム別進捗状況表を使って進捗管理を行った．

進捗の%については，設計工程では設計工程ごとに定めた成果物が完成するごとに25%ずつの完了とし，プログラミング工程では，

- コーディング終了 …… 25%
- 画面または帳表のイメージ確認終了…… 50%
- ファイル更新の確認終了 …… 75%
- 単体テスト表の全項目テスト終了 ……100%

にする等，目安を設けてチェックした．

画面レイアウト

画面名称	振替入金処理		
画面名	GAMEN1	作成日	89年 9月 28日
作成者		(改訂)	年 月 日

```
....*....1....*....2....*....3....*....4....*....5....*....6....*....7....*....8
                                                                    99/99/99
                        ■■ 振替入金処理 ■■

 借方科目     貸方科目              金 額        入金先          入金内容

 銀行 [XXXX]  KKKKKKKK    [999999999] [XXXXXXXXXXXXXX] [XXXXXXXXXXXXXX] [X]
                           [999999999] [XXXXXXXXXXXXXX] [XXXXXXXXXXXXXX] [X]
 A銀行  2001               [999999999] [XXXXXXXXXXXXXX] [XXXXXXXXXXXXXX] [X]
 B銀行  2002
 C銀行  2003

 勘定名        選択 [99]

    (1) A銀行          (4) 仮受金          (7) 受取利息
    (2) B銀行          (5) 仮払金          (8) 雑収入
    (3) C銀行          (6) 本店勘定

                                  確認 [9] (1:OK, 9:NG)
....*....1....*....2....*....3....*....4....*....5....*....6....*....7....*....8
```

図 8 PRODUCEによる画面レイアウト

Fig. 8 Display-layout by PRODUCE

FILE LAYOUT					作成日 89年 9月28日		担当者		ﾍﾟｰｼﾞ 01ﾍﾟ
ファイル名	ファイル名称			装置	ﾎﾞﾘｭｰﾑ	ボリューム名称			ﾌｧｲﾙﾗﾍﾞﾙ
FILE1	下り電文			DK00	@SYSV				FILE
編成	ﾚｺｰﾄﾞ長	ﾌﾞﾛｯｸ	ｷｰ位置	長さ	件数	保存	ファイル容量		
順次	384 B				100	永久	20 S	40,960 B	

項目名	項目名称	位置	型	桁数	要素	内容 〔意味づけ〕
RTZCD	トランザクションCD	1	X	4		
RSPCE	スペース	5	X	2		
RID	電文識別	7	9	1		
RTYPE	端末タイプ	8	X	1		
RN@	端末番号	9	9	2		
RGMN@	画面番号	11	9	3		
RABNK	A銀行	14	X	4		
RBBNK	B銀行	18	X	4		
RCBNK	C銀行	22	X	4		
RKUKIN	仮受金	26	9	10		
RKBKIN	仮払金	36	9	10		
RH@NKN	本店勘定	46	9	12		
RURISK	受取利息	58	9	8		
RZATU	雑収入	66	9	10		
@FILER		76	X	256		
@FILER		332	X	53		

図 9 PRODUCEによるファイル・レイアウト

Fig. 9 File-layout by PRODUCE

表 5　詳細設計進捗管理表

Table 5　Progress management table of detail design

工程	No	進捗チェック日	システム名	地区センタ		運用センタ		支　店	
				予定	実績	予定	実績	予定	実績
プログラム仕様書	1	63.4.8 (金)							
	2	63.4.15 (金)							
	3	63.4.22 (金)							
	4	63.4.28 (金)							
プログラミング	5	63.5.13 (金)							
	6	63.5.20 (金)							
	7								

この進捗状況表は，進捗状況の把握と各担当者の意識を高める上で効果があったと思う．詳細設計進捗管理表を表 5 に示した．

表 5 の進捗管理表は手書きであり，プログラムの本数が増加するに従って大変な作業になってきた．この解決策として，プログラム進捗管理を ECHO-G(リレーショナル・データベース操作言語) を使って行うことにした．

ECHO-G は，パラメタを指定するだけで，データ入力画面プログラムや明細表および集計表が出せる言語であり，プログラム開発の進捗管理には最適であると思われる．

ECHO-G で出力したプログラミング工程の進捗管理表を表 6 に示したが，この他にシステムごとのプログラム本数集計表や担当者別プログラム一覧表，期日指定未完了プログラム一覧表等を，パラメタ指定の変更だけで容易に出力することができる．

プログラム開発の進捗管理は，大規模システム開発ではプログラム本数も多く，開発担当者も多いため，きめ細い管理がむずかしくなってきている．このような状況の中で，ECHO-G を使ったプログラム管理は有効であると思う．

しかし入力作業を必要とするため，今後は開発原始ライブラリから，プログラム ID，言語種別，実績ステップ数等をリストするユーティリティ等と連動して，入力作業を減らし，正確さを増す等の工夫が必要である．

7. システムテスト

システムテストは，システムの品質を確保する上で重要な工程である．

大規模システムの場合，ソフトウェア障害発生時の影響が大きく当システムのような一般ユーザを対象にしているシステムでは，社会的な影響も大きくなってくる．

ソフトウェアの障害を少くするには，設計書の段階からバグの発生しにくいプログラム構造にする等，各工程ごとに品質の確保に努力すべきであるが，最終的に本番後のバグ発生を最少限に食い止めるのは，システムテストの工程である．

今回の開発ではシステムテストを表7の観点から行った．

これらのテストは，本番とまったく同じ機器構成で行うのが最良であるが，ネット

表 6　プログラミング工程進捗管理表

Table 6　Progress management of programming process

サブシステム名　端末

NO	ID	プログラム名	見積数	実績数	名前1	名前2	プ予定日	プ終了日	単予定日	単終了日	優先
353	TMX060	モジュール配信（受信）	0	0	山崎	荒井	04/27	00/00	04/30	00/00	0
266	TMX070	接衛経緯取出し（２２０照会）	100	0	矢野		04/28	00/00	06/19	00/00	3
84	TMX071	折衛経緯ファイル取り出し	0	0			00/00	00/00	00/00	00/00	0
267	TMX080	日付け取り込み（修正）	100	0	矢野		00/00	00/00	06/01	00/90	0
357	TTX012	回線制御（バージョン２）	0	0	山崎	荒井	00/00	00/00	05/20	00/00	0
345	TEX020	営業支援画面再立上げ	0	234	山崎		00/00	00/00	04/28	07/15	0
346	TMX010	日付セット	400	477	山崎	荒井	00/00	00/00	03/30	07/12	0
347	TMX020	端末パワーＯＦＦ送信	500	808	山崎	荒井	00/00	00/00	03/30	07/12	0
348	TMX030	ファイル受信	1500	1508	山崎	荒井	00/00	00/00	03/30	07/12	0
349	TMX031	索引ワークファイル生成	100	80	山崎	荒井	00/00	00/00	03/30	07/12	0
350	TMX032	ジョブＰＯＳＴ発行	5	5	山崎	荒井	00/00	00/00	03/30	07/12	0
351	TMX040	ファイル送信	1000	1014	山崎	荒井	00/00	00/00	03/30	07/12	0
352	TMX050	ファイル送信要求送信	400	528	山崎	荒井	00/00	00/00	03/30	07/12	0
354	TTG030	端末設定ファイル・メンテ	850	865	山崎	荒井	00/00	00/00	03/30	07/12	0
355	TTG050	ＪＯＢ起動指示	300	297	山崎	荒井	00/00	00/00	03/30	03/30	0
356	TTX010	回線制御	1500	2141	山崎	荒井	04/23	07/12	04/22	00/50	0
358	TTX020	ＩＰＬ後ＪＯＢ起動	200	218	山崎	荒井	00/00	00/00	03/30	07/12	0
359	TTX040	ＪＯＢ強制終了	100	110	山崎	荒井	00/00	00/00	03/30	07/12	0
85	TTX060	ＬＣＰ再立ち上げ	0	139			00/00	00/00	00/00	07/12	0
360	TUX020	受調画面再立上げ	0	139	山崎		00/00	00/00	04/28	07/12	0
142		日次帳表出力処理	0	0	佐原		00/00	00/00	00/00	00/00	0
143		要報告一覧出力	0	0	佐原		00/00	00/00	00/00	00/00	0
合計			7055	8563							

表7　テスト内容一覧表

Table 7　Contents of tests

テスト項目	テスト内容
業務テスト	基本業務が各システムを通して正しく動くか．
ネットテスト	地区センタ相互間での入出力テスト（分散しているデータベース相互間の業務テスト）
結合テスト	他のシステムとの結合が正しく行われるか．
性能テスト	各トランザクションごとのＣＰＵ処理時間，端末の応答時間，地区センタのスループットテスト
ラッシュ・テスト	処理がラッシュした時にも，システムが正常に稼働するか．
リカバリ・テスト	障害発生時のリカバリが手順通り行われるか（ＨＷの二重系に対する切り替えテストを含む）
移行テスト	旧システムから新システムへの移行を行い，業務が正しく稼働するか．
新・旧混在テスト	新システムと旧システムの双方向からの入出力テスト（分散システムであり，新システムへの移行を段階的に行うための，新システムと旧システムとの共存時のテスト）

ワーク網や端末の台数等，現実には本番環境の準備がしにくい場合がある．

これらの問題については，端末ドライバの作成やテストデータ・ジェネレータにより対処した．

1) 端末ドライバ……端末ドライバは，支店システムであるシリーズ8/モデル100 IIの業務画面からの入出力に替わり，支店システムへ前もって入力したトランザクションを，あたかも端末業務画面から連続して入力しているかのようにシミュレートする機能であり，各業務の単体性能を評価する時に効果を発揮した．

2) リラン機能によるシステムテスト……システムテストを行う場合，本番に合ったテストデータでテストをすることが重要であり，今回のシステム開発のように旧システムが存在する時は，旧システムのデータでテストすることは必須になってくる．

このため，今回の開発では，TP（トランザクション処理システム）にあるリラン機能*をシステムテストのために使用することにした．

仕組みは，まず旧システムの1日の業務サイクルを動かす前にマスタファイルを保存し，一日の業務サイクル終了時に，取引ログと，更新後のマスタファイルを保存する．次に新システムにて，更新前のマスタを新データベースに変換し，取引ログを共通ジャーナルファイルに変換後，TPのリラン機能を動かし，リラン後の新システムのデータベースと，旧システムの更新後のデータベースを比べることによってプログラムの機能チェックを行う．

これらの一連のテストツールを作った理由は，業務テストだけでなく，性能テスト，ラッシュテストでも使用するためであった．

オフコンの場合，オンライン・システムの検証用ツールはまだまだ少く，テストデータの自動作成—テストの自動実行—実行結果の自動照合—テスト結果の自動作成等，テストの自動化の実現が望まれる．

8. システム開発の評価

システム開発終了後に，目標に対する達成度の評価を行うことは，開発から稼働に移ったシステムを改善することや，次のシステム開発をより正確にするために必要なことである．評価はシステムの品質評価から行った．達成度を以下に記す．

- 機能達成度　（要求定義に書かれた機能の評価）　90%
- 操作性達成度　（システムの使いやすさの評価）　90%
- 性能達成度　（各システムの目標に対する性能評価）　50%
- 信頼性達成度　（障害対策を含めたリカバリ評価）　70%
- 適用性達成度　（他システムとの整合性の評価）　90%
- 拡張性達成度　（システム変更の容易性の評価）　80%

以上のように機能要求は達成したものの，性能問題および信頼性については満足できる評価とはいえず，システム稼働後に改善への努力を行うことになった．

次に開発見積りとスケジュールに対する評価であるが，運用システムおよび支店シ

*リラン機能：オンライン稼働中の入出力データを共通ジャーナルエリアに保存しておき，システムダウン時にこの共通ジャーナルエリアからリカバリする機能．

ステムは予定の工数で開発を終了したが，地区センタ・システムでは，プログラミングおよびシステムテストの工程での見積り差違が発生した．これは，地区センタ・システムが，リアルタイム処理プロセッサ（TP）を初めて使ったことと，分散データベース間の処理機能の複雑さによるものであった．

次回の開発では，SEに対するTPシステムの教育により，コーディング基準およびテスト方法の標準化を行い，品質および生産性を高めたい．また必要ならば，開発生産性見積り基準の見直しも行いたい．

この他の評価項目として，工程別の誤り（バグや仕様書の記述ミス）を集計し，それぞれの発生原因を追求し，問題のある工程を探し出す等のバグ原因評価も必要である．このバグ原因評価については，今回の開発ではシステムテストでのバグ発生件数を集計したに止まってしまった．これはシステム開発が始まる時に，あらかじめ工程別誤り発生件数を想定し，管理目標を決め開発工程ごとにデータを収集しなかったためであり，今回のプロジェクトの反省材料となった．

9. おわりに

以上，オフコンにおける大規模開発プロジェクトの開発過程を述べてきたがシステム開発と言う意味では，汎用機であろうとオフコンであろうと，同じようなプロジェクト管理が必要である．ただ，工数見積り時の生産性見積り基準の数値等の基準数値が違うだけである．

今回のシステム開発も，オフコンを意識したプロジェクト管理を行ったわけではなく，一般的なプロジェクト管理の方法によったが，3.2節で述べた開発作業ブロック図や，6章のECHO-Gを使ったプログラム進捗管理等に工夫をこらした．

今回のプロジェクトでは，開発を進めていく中で進捗の遅れ，システムテストの方法等，問題点が発生の都度，解決策を取ってきたが，次回のプロジェクトではこれらの項目を開発工程表に盛り込みたいと思う．

参考文献
［1］ 管野孝男，ソフトウェア開発のマネージメント，ソレカラ社，1987年4月．
［2］ 宮田弘之介，工事工程管理―工程計画の立て方と利用法，鹿島出版会，1985年8月．
［3］ 石井康男，ソフトウェアの検査と品質保証，日科技連出版社，1988年2月．
［4］ シリーズ8 DPS 10トランザクション処理システム解説書，日本ユニシス，資料コード481755711-O，1986年11月．

執筆者紹介 市 川 常 男(Tsuneo Ichikawa)
昭和24年生．46年工学院大学電子工学科卒業．同年，日本ユニシス(株)入社．57年よりオフィスコンピュータ金融系システム開発に従事．現在，BSシステム本部OAシステム一部に所属．

大規模小売業の統合情報システム構築事例

A Sample of Building an Integrated Information System for Large-scale Retailers

福 原 俊 作

要 約 大規模小売業における情報システム化は，従来の単一情報システム構築から「多品種少量販売による新たな利益の創造」と「規模の利益の取り戻し」という相反する目的を達成する方向にあり，システム規模の拡大と関連分野の広がり・複雑化を招いている．

ここに紹介する構築事例は，新規開発と既存システムとの改造が混在した総工数3000人月規模の統合情報システム開発事例である．

本稿では，開発システムの内容を紹介するとともに，急激な情報システム開発の大波への対応と，中小規模の個別情報システム開発が同時に進行し全体として大規模化しているシステム開発の構築環境整備に焦点を当て，その工夫点等を紹介している．

特異な技法・ツールの導入，工夫が求められるものではなく，この種の開発では，個々人の仕事を見えるようにすることと，類似処理例が多くこれらをいかにパターン化し汎用化を図り浸透を図っていくかに，生産性と品質の向上がかかっていると実感している．

Abstract The computerized information system for large-scale retailers is now shifting from the conventional construction of single-purpose information systems to the building of integrated multi-purpose information systems aimed at 'creation of new profits from the sale of commodities on a small-lot/large-variety basis' and 'return of economies of scale', thus leading to the expanded scale of systems and the more complicated coverage of related areas.

The sample of systems construction described here is about the new development of an integrated information system which has required a 3,000 man-month workforce to intermix newly-developed systems with enhanced existing systems …just a case in point on the way toward integration.

Besides touching on what the targeted information system is like, this paper discusses how to react to the rapidly growing surge of information systems development, focusing on the environment where systems building, as a whole, is getting larger and larger in scale with the parallel development of small-and medium-sized individual information systems under way at the same time.

The author's realization is that this type of development does not call for any new particular techniques or development tools but depends on making individual developers' stints visible, the standardizing of many similar existing processing patterns and efforts to make wide use of those standardized processing procedures for higher productivity and quality.

1. はじめに

今，ナショナルチェーンを展開している大手GMS* の情報システム化にはめざましいものがある．

たとえば，日経コンピュータ[4]に紹介されているダイエーのマス・マーチャンダイジ

* GMS(General Marchandise Store)：食料品・衣料品・生活関連用品等のすべてを品揃え，大規模小売り業といわれる大手チェーンストアの多くはこのカテゴリに入る．

ングの確立を狙った総合システム化計画（新POSシステムのみで110億円投資），IY社の業務改革の継続（POSの初期投資のみで100億円），西友の経営基盤の整備（SMILE-NETに対する投資120～130億円），ジャスコの営業効率化推進（150億円）等の構築例である．

これは，情報システム*を核とした「少ない在庫・経営資源で多品種少量販売を実現」し，「消費者の多様化・個性化・潜在化するニーズ・ウォントを的確にとらえ企業活動に活かし，これに働きかけ新たな利益を創造」するための変革であり，スケールメリットの取り戻しであり，変化への機敏な対応であり，優位性・独自性の確立であると言える．

ここに紹介するGMSであるI社の統合情報システム構築も，まさにそのものであり，1990年度完了予定の総投資80億円規模の情報システム開発事例である．

この統合情報システムに着手以前は外部への委託はほとんどなく，1000人月/年規模のシステム化を受け入れるだけの土壌が準備できていなかった．したがってシステムを構築すると同時に，構築環境を整備することが当社の大きな役割であった．

技法の導入，各種標準化，協力会社（パートナ）管理，体制の整備，品質の評価，高生産性のための各種試み，ツールの用意等手掛けてきたことは多い．これらの全体像を紹介すると同時に，その中でもとくに重要と思われる事項について説明を加えている．

2. I社の統合情報システム

2.1 情報システムの領域と方向性

I社は，表1の事業規模を持つ大規模小売業者である．

表1 I社の事業概要
Table 1 Business outlines

本部所在地	大 阪 市
事業内容	衣料品，食料品，電器，家具，レジャー用品，日用雑貨等の総合小売業のチェーンストア
設 立	昭和24年12月（創業 大正10年5月）
資本金	104億400万円（昭和63年2月現在）
売上高	3194億円（昭和62年実績）
店舗数	60店舗（昭和63年4月現在）

I社の統合情報システムは，本部・店舗・配送センタ・取引先の双方向オンラインネットワークシステム完成期の1987年より構想に入り，1991年2月の完成を目途に開発中である．情報処理システム開発のみで3か年総工数3000人月強のプロジェクトであり，現在，個別情報システムが段階的に順次本番稼働を始めている．

本部・店頭現場の判断のずれの是正，取引先・配送センタとの情報ギャップの是正，経営指標管理の基盤強化，人時生産性**の向上を狙っており，「情報の共有化・即時化」を目的としている．

---

* 情報システム：情報システムを「人」と「仕組み（組織・制度・手続・道具等）」と「情報処理（EDP）システム」から成るものととらえている．

** 人時生産性：小売業でよく使われる指標であり，単位時間当たりの生産性（たとえば売上高）を意味する．

一般的には，業務系システムと情報系システムの切り口から語られるが，本システムは図1のように商品系と管理系システムの切り口でとらえられている．各々は，商品管理・販売系の営業活動を主体としたシステムと，経営資源としての人・物・金・情報を積極的に活用していく経営管理系のシステムから構成されている．

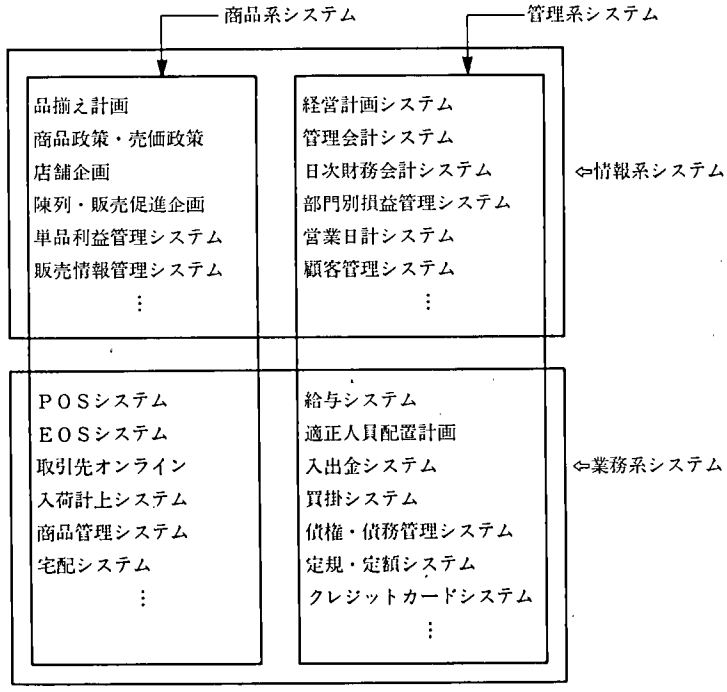


図 1　商品系情報システムと管理系情報システム

Fig. 1　Business information system and management information system

2.2　情報システム構築と基盤整備

基幹情報システムの再整備も大きな領域を占めている．たとえば，商品系の大きな柱に POS システムがあるが，前準備として商品コード体系の再設定が必須である．単品管理のコード化であり，従来の商品管理のみのコード体系から販売管理を含めた管理体系への移行である．

さらに，データの同期化と捕捉基準・精度の統一化がある．従来 EOS*が先行しており，発注状況は当日締め時に把握できていたが，仕入・販売状況が 1〜2 日遅れで収集されていたため情報としての価値が半減していた．ところが，販売情報が単品で即時把握できるようになると入荷計上情報も即時把握が必須となる．売れ具合を評価しアクションに結び付けるには，当日発注・入荷予定情報/実績情報，配送センタ・店舗の在庫情報が必要となる．このことは商品外の人，物（什器，設備・売り場・フェース等），金（売上高，経費，仕入高等）に関しても言える．販売情報と売場要員（人時効率），売場面積（売場効率），資金（回転差資金効率）等との関連把握である．

捕捉基準・精度の統一には，売上種別（現金・ローン・カード・ギフト），売上計上

* EOS(Electronic Ordering System)：コンピュータを利用した補充発注システム，また本部・店舗・仕入先との間で受発注データをオンラインで交換するシステム．

(金額決済・一部入金・前金・代引)，在庫（格下げ・格上げ・廃棄・ロス)，目標基準（ロス率・人時生産性）等の再規定と取り扱いがある．

これらは，情報活用の基盤整備そのものであり，既存情報処理システムの改造を必須としている．当社系ホストで稼働している業務系プログラムのうち，商品系 1700 本，管理系 2200 本，その他 600 本，合計 4500 本の半数近く約 2000 本のプログラムが改造・統廃合の対象となっている．

2.3 取り組み計画と概要

全社情報システムとして課題領域の明確化と絞り込みが行われ，その後大きく商品系情報システムと管理系情報システムの 2 系列でシステム構築が推進されている．

大日程計画概要を図 2 に示す．

2.3.1 商品系情報システム

商品系情報システムは，POS システムの導入・展開を第一優先としている．その上に，店舗運営・単品販売管理等の新規システムが構築される．

既存の商品管理（生鮮・定番・衣料・生活関連商品)，特売・大型商品宅配・EOS・取引先オンライン・入荷計上等のシステムをベースとした機能追加型のシステム構築が主体となっている．商品コード体系の再整備と管理体系の変更が進められており，これに伴う既存情報処理システムの改造も大きなウエイトを占めている．

2.3.2 管理系情報システム

スタッフ部門の業務支援と同時に人・物・金に係わる情報の把握・管理（データベース化）を大きな命題としている．これは，計数管理の土台を固め，経営情報システムへの展開を意図したものである．

取り組み領域には，会計（債権債務・財務会計・管理会計・入出金管理・資金管理・固定資産・関連会社等)，人事（人時効率・要員管理・配置異動・勤怠等)，予算（予算体系・編成・制御・評価等)，資産（店舗開発・設備什器・レイアウト・契約等）とこれらの管理事務があり，現在，本部を主体とした業務系システムの一次システム構築完了段階に入っている．今後，店舗管理系情報システムの構築と同時に，商品系情報システムと連動した情報系システムの構築が主体となってくる．

3. I社開発プロジェクトと当社体制

I 社のコンピュータネットワーク体系は図 3 のようになっており，本部基幹情報システムを担当する当社系システムと商品販売管理を主体とする IBM 系システムと店舗管理を主体とする日立系システムの 3 系列システムから構成されている．

客先プロジェクトは商品系・管理系・店舗系の 3 系列から構成されており，各々のプロジェクトに各メーカ系のシステム開発プロジェクトが参画する形態をとっている．当社開発プロジェクトは，商品系情報システムの商品マスタ管理・受発注・特売・大型商品宅配システムの改造と管理系情報システム・顧客系情報システムの全般を担当している．開発期間は 1987 年下期～1991 年 2 月までの 3.5 年，開発規模は 1400 人月強（企画・立案と情報処理システム開発の総計）のプロジェクトである．

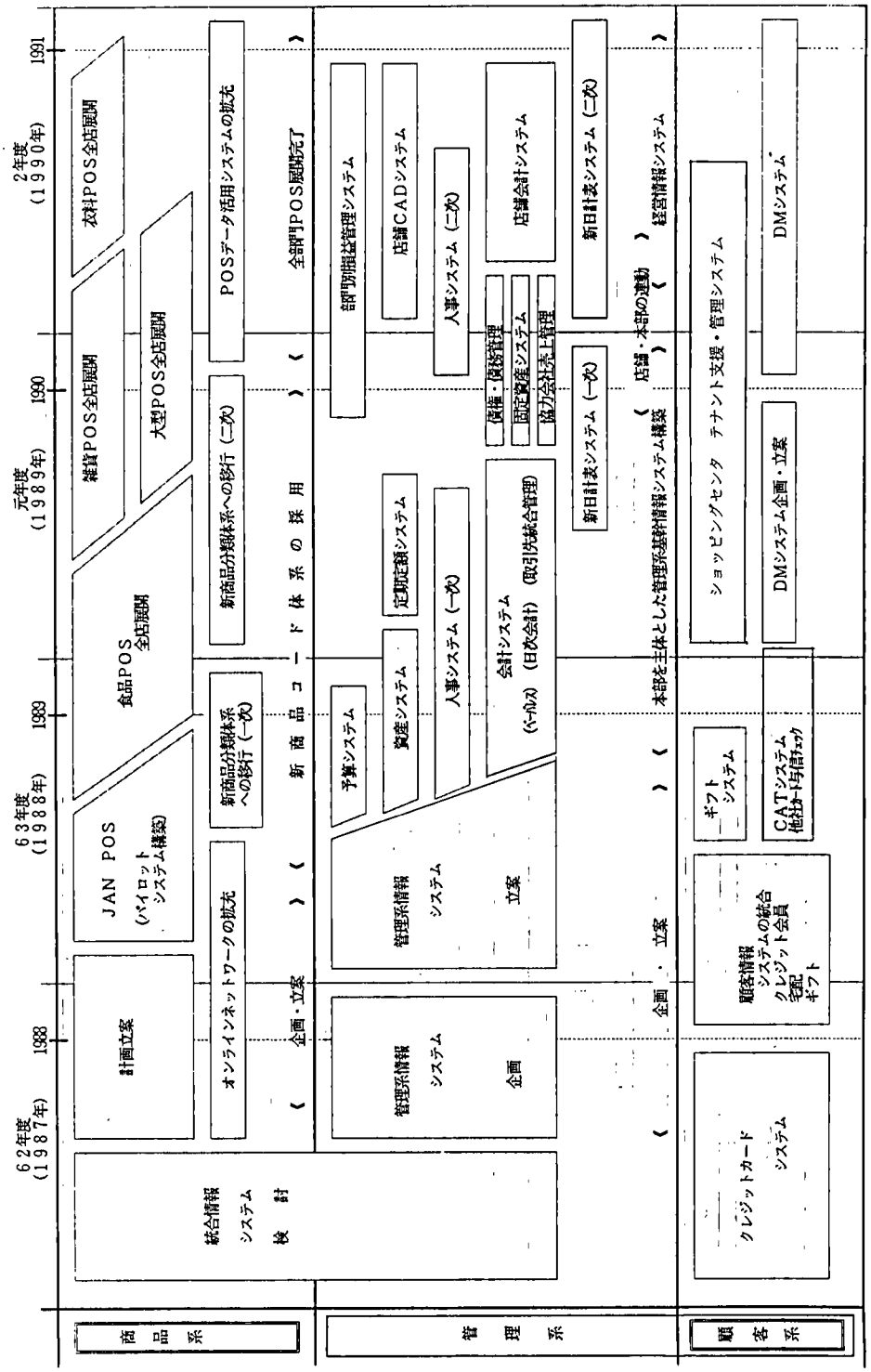


図 2　大日程計画

Fig. 2　Planning of the development

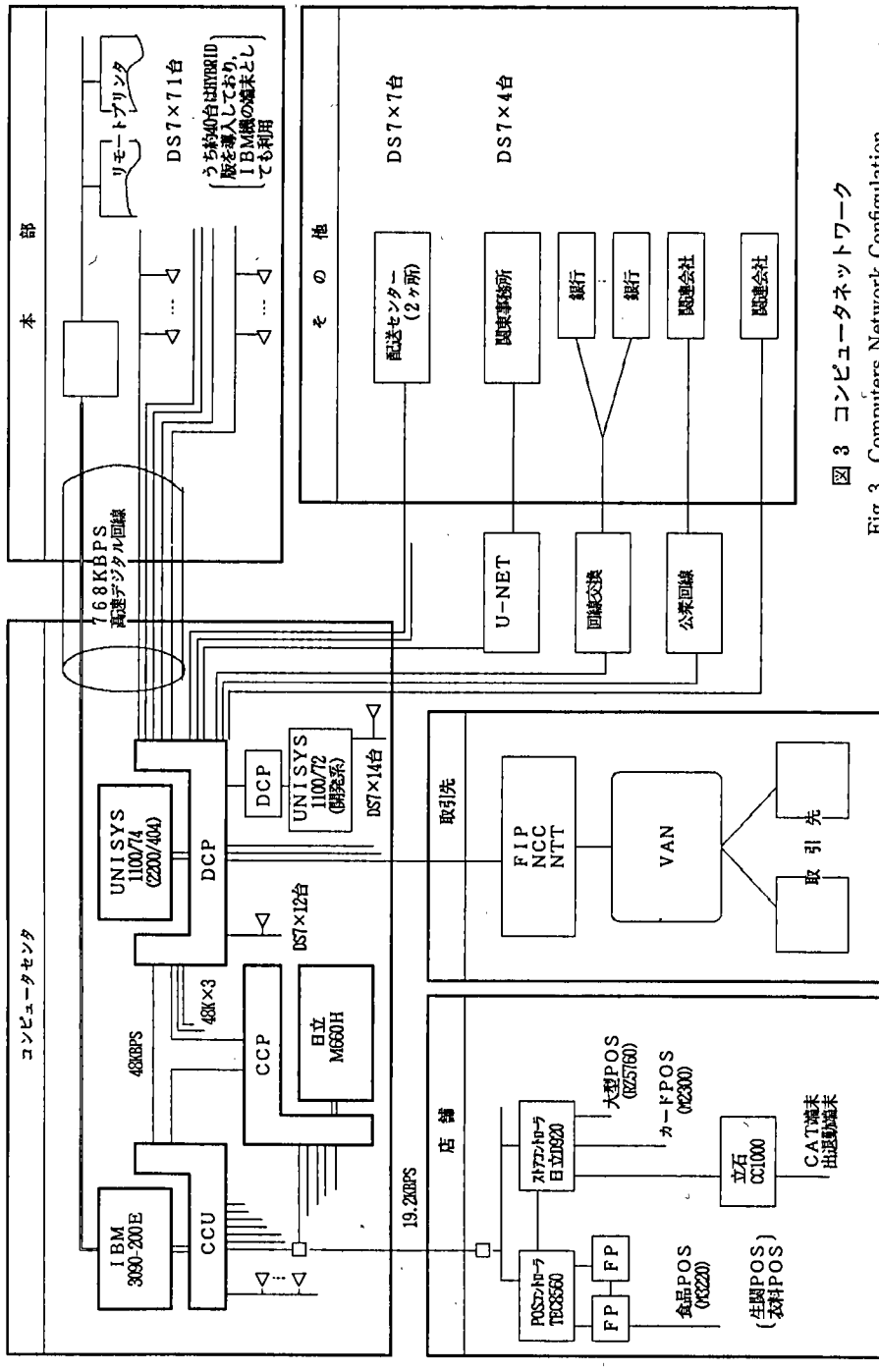


図 3 コンピュータネットワーク

Fig. 3 Computers Network Configulation

3.1　システム開発環境と当社開発プロジェクトの特徴

3.1.1　システム開発基盤の整備

開発プロジェクト発足当時の客先情報システム担当要員は20数名であった．

前年度までの情報システム開発は150～250人月/年間程度であり，クレジットカードシステム，大型商品宅配システム等の新規システム開発はあったものの，データをコンピュータにのせることと，コミュニケーション強化を狙ったコンピュータネットワークシステムの強化が主体であり，ようやく情報処理システムの利用者部門への浸透が始まった段階であった．そこに，全体で800～1000人月/年間規模の情報システム開発が開始されることになったわけである．

大規模・長期間にわたるシステム開発は，利用者部門との連携，推進体制，構築稼働後の運用・保守・受け入れ体制の整備とプロジェクトマネジメント，システム開発技法の導入と定着等が重要である．これらを従前以上に確実に浸透させなければならなかった．そこで，システム開発と同時に従来からの標準化推進を強化し，必要の都度，形態を整える方式で情報システム構築基盤の整備も進めることにした．

3.1.2　多様な情報システム開発と開発要員対応

予算措置上の開発工数は設定されているが，大枠の構想があるだけで，個別開発計画と年次開発計画の立案から作業を開始しなければならない*．しかも既存の情報システムを活かし新たな情報システムを追加・結合していく形態がとられているため，会計システム・人事システムのように核になる部分は数百人月規模のシステム開発となるが，それ以外は数十人月単位のサブシステムごとの開発・改造が主体となる．また，半期ごとの商品系情報システム改造作業も，70～100人月/3～4か月での対応が必須となる．これらとは別に既存システムの保守・改造も発生する．

このように，性格の異なったシステム開発が入り混じっており，担当要員も長期張り付け型ではなく，都度対応が必須であり，要員配分とスケジュール調整がとくに重要となる．

3.1.3　厳しい稼働環境下でのシステム開発

コンピュータを停止できるのは，元旦と年2回の棚卸日のみであり，24時間年中無休の状態である．夜中に前日最終分の取引先発注を行い，その後，締め処理を行う．さらに，当日分の売価変更・発注台帳変更・インハウスカードのブラック情報等の送信準備・全店送信にかかり，早朝になると配送センタの入荷計上・出荷が始まる．昼間は，カードの与信チェック・大型商品の宅配予約・店発注・取引先発注・営業日計チェック等のオンラインリアルタイムシステムが主体となる．閉店日はあるものの，全店一斉とはならずコンピュータ側からすると休まる暇がない．

また表2のように大量のデータを扱っており，これらのデータ処理過程での誤り修正には膨大な時間が必要であり，エラー発生は経営上致命的となる．

3.1.4　当社要員体制

客先システム開発に参画している当社プロジェクト要員は，当社要員4名，客先常駐の協力会社要員8～10名，都度対応の協力会社要員10～40名の計25～50名であ

---

*　I社のシステム開発の予算処理は，計画稟議と実施稟議からなっており，投資枠内で，都度の個別情報システムの実施計画を策定する形態をとっている．

表2 主なデータ取扱量
Table 2 Main data volume

取り扱い商品品目数	常時40万点
発 注 量	伝票換算で4～5万枚/月次 20～25万品目/月次
売上点数	2000万単品/週次
客点数(レジ集計回数)	270万/週次

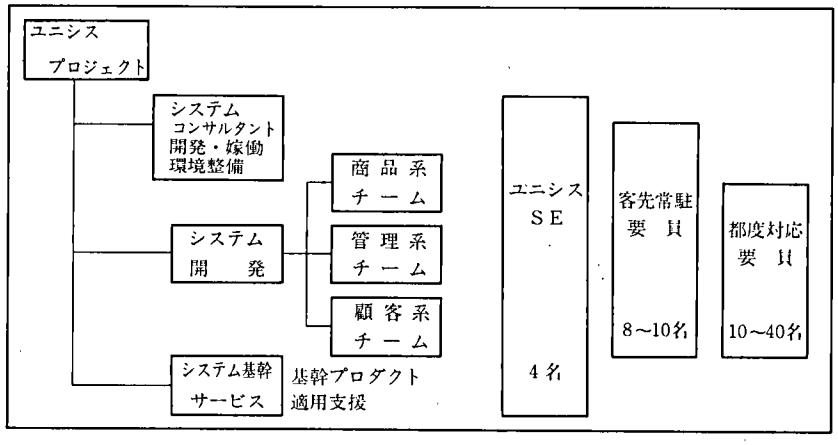


図4 当社体制
Fig. 4 UNISYS project formation

り，図4のような開発チームを編成している．

客先常駐協力会社要員は，ソフトウェア対応主体のSE，PE(プログラマ)と，業務対応主体のSEから成っている．ソフトウェア対応SE，PEは共有部分の設計やプログラミングと既存システムの保守，委託先への技術指導等を行っている．

業務対応SEは，新規システム開発の企画・立案*開始時点から参画しているSEであり，あらかじめ設定した製作工程の委託予定会社のSEを担当としてあてている．担当システムが本番稼働し，成果物を納品するまで製作工程の責任者として位置付けている．

3.2 開発工程と開発プロジェクトの役割分担

システム開発工程は図5に示すように情報システムの企画・立案を主体とした前工程と情報処理システムの設計までの中工程，製作・移行・運用の後工程に分割した考え方を採用した．

当社プロジェクトは，この開発工程にそって，前工程主体の「システム企画・立案プロジェクト」と中工程・後工程担当の「個別情報システム開発チーム」の体制をしいている．

これは，情報システムの企画・立案内容が以降の作業とシステムの完成度を大きく左右することから，客先との共同作業による企画・立案を重視したことと，中工程からは情報処理システム開発と運用設計・利用者部門教育・操作マニュアル作成等の作業(EDP外システム構築)とを分離し，製作主体のチームを発足させ高生産性を重視

* 企画・立案：何をするかを明確にし，それをどのような資源配分で実現するかを立案すること．

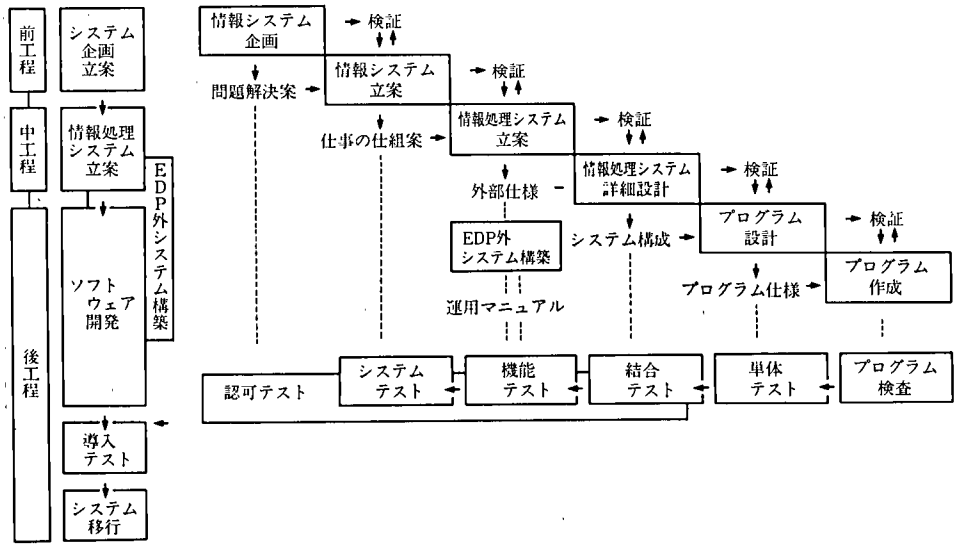
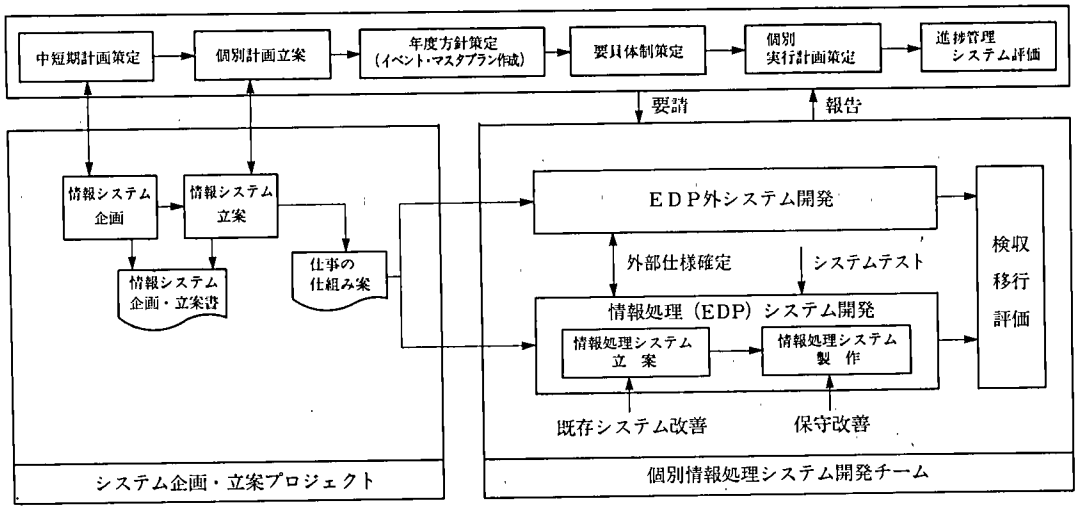


図 5　開発工程概念図

Fig. 5　Process of the system development



企画：何をするかを明確にする
(取り組み課題の明確化と問題解決案の策定)

立案：どのようにするかを明確にする
(資源配分の決定と仕事の仕組み案策定)

図 6　企画・立案プロジェクトと開発チーム

Fig. 6　System planning project's and development team

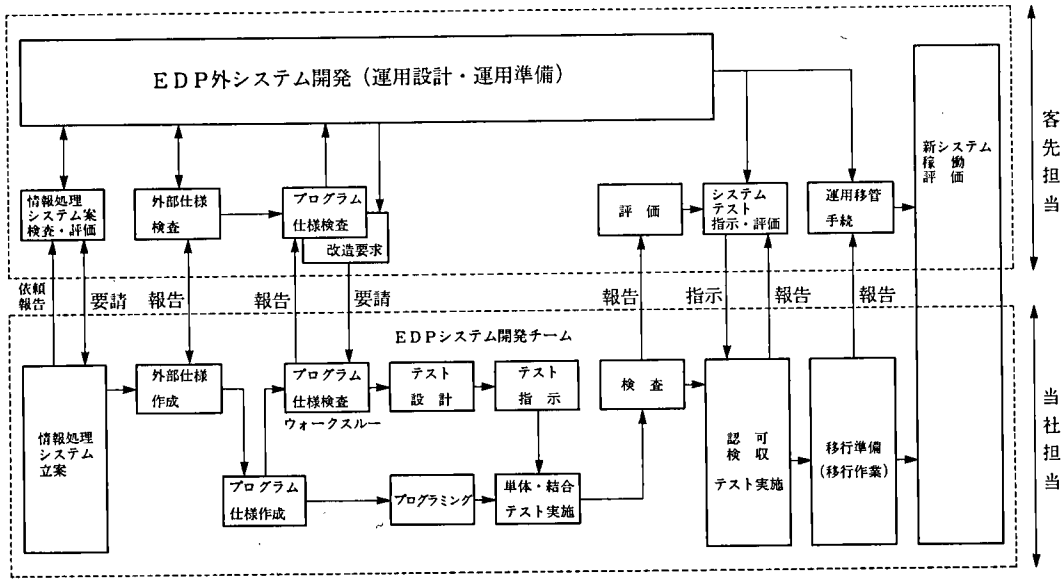


図 7　情報処理システム開発における役割分担

Fig. 7　Assignment on the system development

したことによる．

図 6 に企画・立案プロジェクトと開発チームとの関連，図 7 に開発チームの構成を示す．

3.3　システム開発状況

現在，50 名強の当社プロジェクト要員によるシステム開発の最盛期にあり，順次個別情報システムの導入をむかえている．

主な構築システムとして，商品系システムでは，新商品コード体系への変換を手掛け，POS 展開の推進がある．

管理系システムでは，270 人月・新規プログラム約 670 本・新規分総ステップ約 265,000 (COBOL プログラム，プロシージャステップ) の会計一次システム，人事一次システムの構築がある．

また顧客系システムでも，顧客管理とギフト管理を，約 30 人月・85 本・20,000 ステップで完成させている．

単にシステム開発に留まらず，開発環境の整備（開発工程の整備，企画・立案技法の導入，開発推進・見積り・支援ツール・システム検収・運用の標準化，利用者部門を含めた開発推進体制の整備等）にも力を入れており，情報システム構築基盤の確立に大きく貢献している．

4.　当プロジェクトで採用した技法・工夫点

システムの開発に先立ち，あるいは開発の過程で各種技法の採用・工夫を試みてきた．これらを標準化の観点から眺めると図 8 のように分類できる．

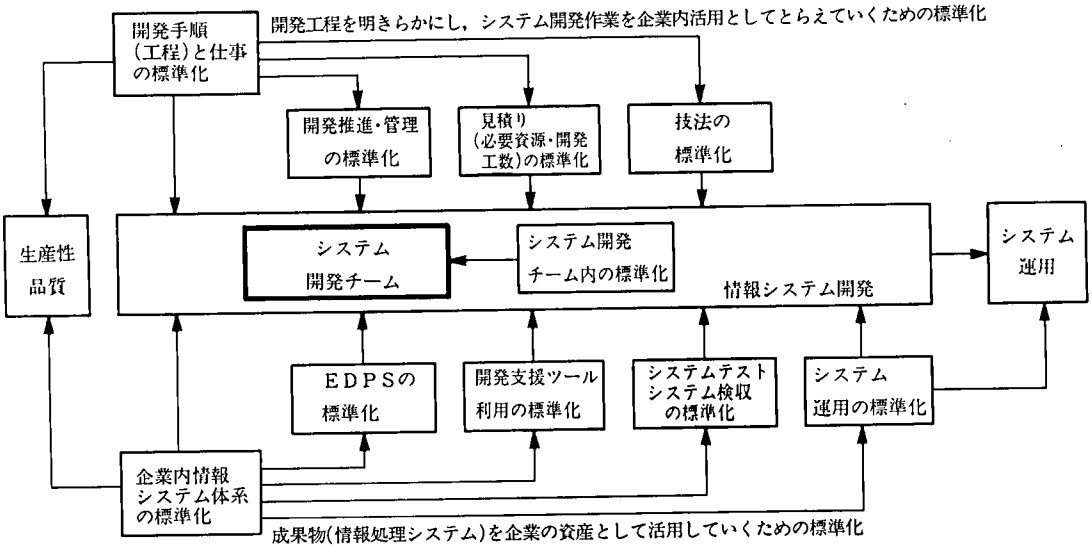


図 8　情報システム開発の標準化体系

Fig. 8　Standardization item of the system development

また，開発工程別に採用した技法・工夫点等をプロジェクトマネジメント，協力会社（パートナ）管理，その他の観点から要約すると図9のようになる．これらに費やした当社の工数は，約14人月/3か年である．

4.1　採用した技法・工夫点の重点事項

システム開発環境の整備として推進してきたこれらの事項は，具体的には考え方としてまとめたもの，実際の活動様式としてまとめたもの，プログラム仕様書としてまとめたもの，各種規程集としてまとめたもの，ワークシート類としてまとめたもの，定例会議の形態等から成っている．

環境と整備の過程により，重点施策事項は異なるが，I社では

1)　利用者部門主体型の企画・立案段階を確立すること，

2)　特別な技術に依存しなくても，企業の情報システムを機能面から分解すると類似機能が多々あり，この特性に着目し生産性と品質の向上を図ること，

3)　同じような試みを各自の経験則ベースで繰り返す無駄を排除すること，

4)　計画と実績をある一定ルールのもとで把握し，見える形にし，数値情報をたくわえ指標として活用していくこと，

に重点を置いてきている．

以下にこの観点に基づき，主な事項の説明を加えていく．

4.2　当プロジェクトで採用した主な技法・工夫点

4.2.1　アプリケーションレベルプロトコルの規定

オンラインリアルタイムシステムでは，障害対策・機密保持・入出力先制御・経路制御等の独自の設計考慮が必要となる．また，メニュ制御等共有機能も多い．これらの繁雑さから解放し，業務処理プログラムではトランザクション（あるいはデータファイル）処理のみに専念できるようにしたのが，アプリケーションレベルプロトコル

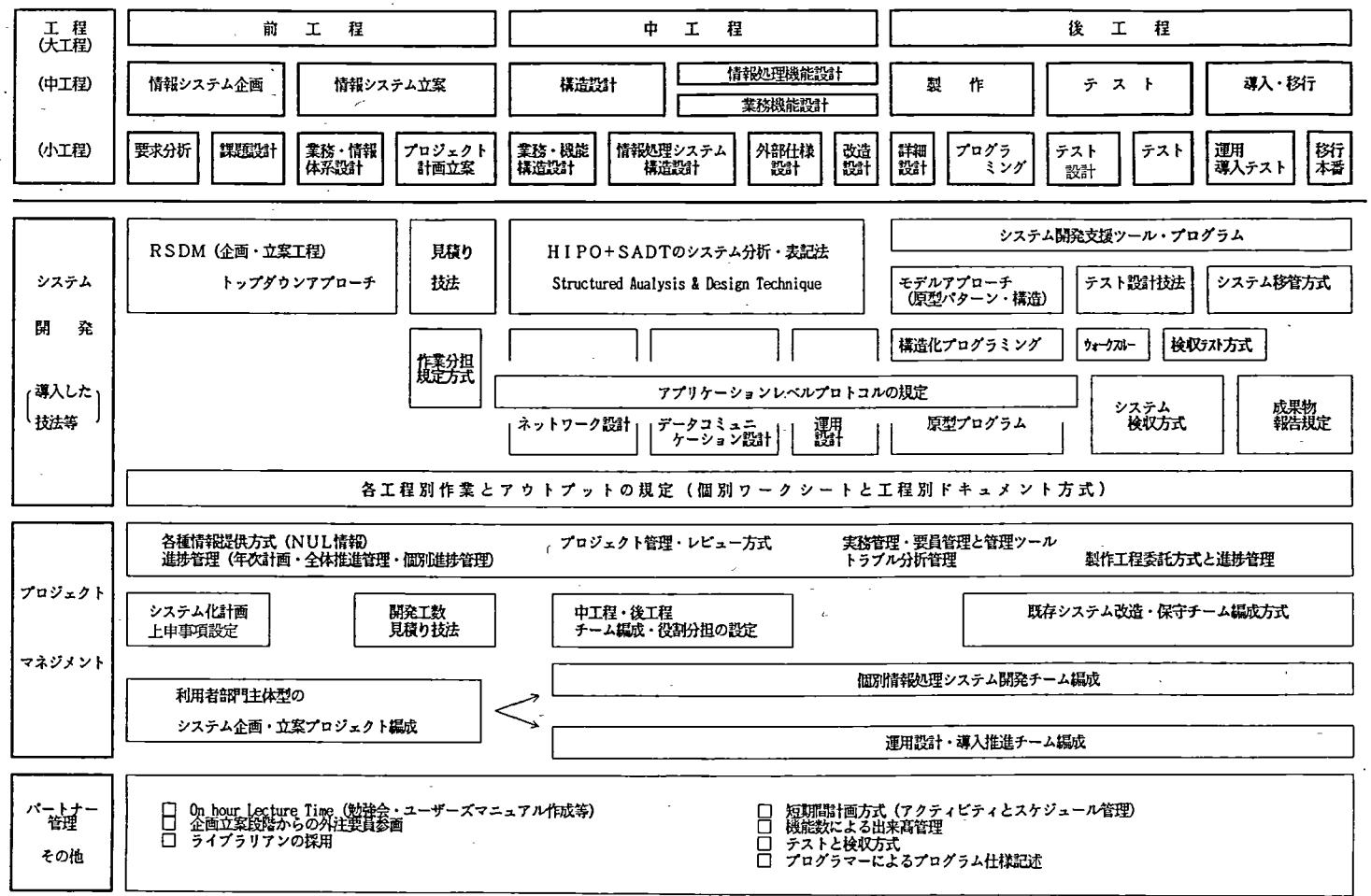


図 9 導入した技法・工夫点

Fig. 9 Technical method and device

の規定である（図10）.

第一版では会話型処理を主体に，ホスト・端末の機能分担を規定し，マンマシンインタフェースを更新・変更・照会・検索・削除の基本機能との関連で示している.

これをベースにオンラインシステムの機能を「会話型処理制御」「メニュ構成」「業務処理制御」「ファンクションキーの利用」「機密保持機能」「二重入力制御」「参照画面の出力形態」「障害対策」「ネットワーク監視」「システム運行」等の面から規定し，これを支援するサービスプログラム群（入力制御・メニュ制御・出力制御・ネットワーク監視・終了/エラー処理等）を用意し，業務処理の基本パターンとプログラム構造を設定した.

これら以外に画面構造・メッセージ構成・テスト方式・各種テーブル類の規定を行い，業務処理プログラムではトランザクション処理のみに専念すればよいようにしている.

その後，コンピュータ間のファイル転送/会話型処理規定・会話型処理の直接入力方式規定・帳表/画面出力の運用規定・帳表作成基準の設定・スケジュール管理・テスト方式の機能強化等を行い，現在に至っている.

アプリケーションプロトコル規定集としてまとめるだけでなく，コーディングパターン集も作成しており，初心者のリアルタイムプログラム作成に大きな成果をあげている.

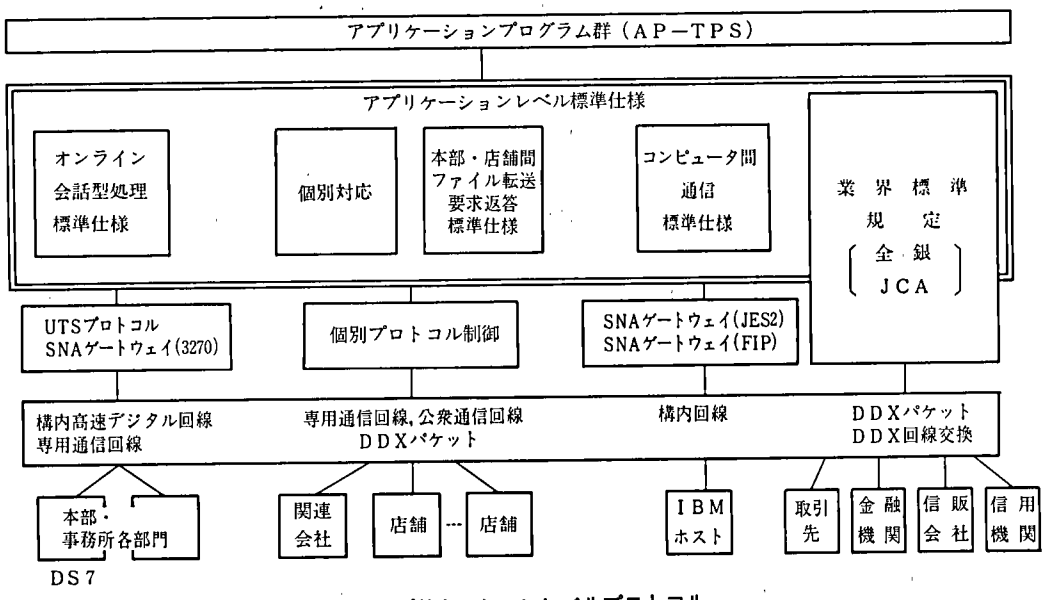


図 10　アプリケーションレベルプロトコル

Fig. 10　Application level protocol

4.2.2　原型プログラムの活用

これはプログラム製作工程で最も効果をあげている方式である.

前項のアプリケーションレベルプロトコルの規定により，本部会話型処理，店舗とのトランザクション転送等の基本型ごとに業務処理プログラムの入出力基本パターンが定まる．これを原型にプログラムの基本構造を作りあげると，後は業務処理の特性

のみをサブモジュールとして付加するのみで，業務処理プログラムは完成する．

さらに，この原型プログラムをベースに業務処理の特性を考慮した更新型・参照型・検索型・変更型・削除型の業務処理基本構造プログラムを作成すると，各々のプログラムは入出力画面・編集・データベースアクセス・処理ロジックに依存する部分だけになり，大幅な簡略化を図ることができる．プログラム構造部のテストも不要になり，効果は大きい．

業務処理プログラムの処理部分は通常 400～800 ステップ程度であり，そのうち原型部が半分近くの 200～300 ステップを占めている．図 11 に原型プログラムの概念構造図を示す．

4.2.3 ソフトウェアテスト技法の採用とテスト計画工程の導入

これは，テストの生産性向上と品質の確保を狙い導入したものである．

テスト・検証・デバッグの分離，開発工程に連動したテスト工程とテスト基準の設定，実施方式とテスト完了基準の設定，テスト工程とテスト内容（各工程での成果物との相違点の発見方式）の設定等を行い，さらに，開発工程を規定し（図 5 の開発工程概念図の下段にテスト工程を設定している），テスト計画書（テスト指示書）に基づき実施し評価できるようにしたものである．

基本的には，テスト工程前にウォークスルーを実施するようにしたが，現在ウォークスルーは強制せず，業務処理原型プログラム作成時点・複雑なプログラム，あるいは要員のプロジェクト参加時点に OJT として体験させる程度にとどめている．

テスト計画技法導入時点での実績値では表 3 のようにウォークスルーの成果が大であるが，次に示す理由により必須としていない．

1) ウォークスルーで誤りのパターンを見つけると次回からエラー発見件数は極端に減ること．
2) 前工程・中工程のエラー発見のためテストはいずれにしても実施すること．
3) 検収用のテストも必要であること．
4) 原型プログラムのウォークスルーが終わっていれば，製作工程でのエラーはプログラム仕様書と単体テスト指示書での発見率が高いこと．
5) プログラム標準化に関する事項は，週次のレクチャータイム・ユーザーズマニュアルを通じて習得できること．
6) ウォークスルーを実施すると 2 人日/プログラム（5 人×3 時間）の工数が必要なこと，等．

テスト計画・実施は大きく 2 段階に分け，内部構造・結合テストまでは製作チームの作業分担とし，機能・システムテストは検収者がテスト計画を立案し，EDP 外システムの設計/企画・立案との相違点の発見テストと位置付けている．これらは，図 7 の開発チームとの役割分担で述べている通りである．

4.2.4 ワークシート類の活用

その工程の考え方・進め方を具現化したものとしてアウトプットを規定し，工程別に分類したワークシートを一式用意している．

流用しているもの，新たに設定したものがあり，必要に応じ追加・改造を加え現在に至っている．前工程ではシステム化の性格により，帰納的アプローチと演習的アプ

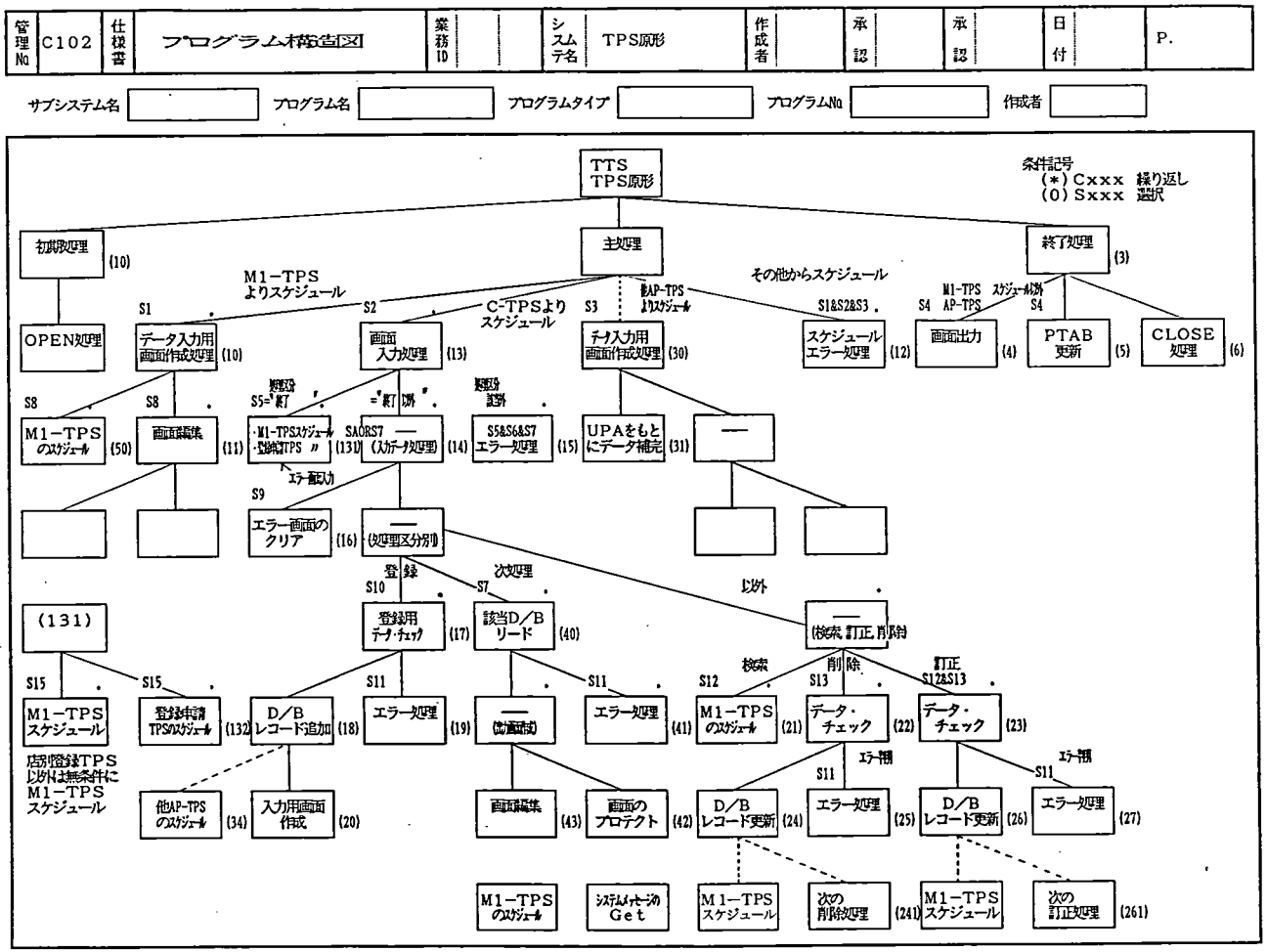


図 11　原型プログラム構造（例）

Fig. 11　Structure of the original program

表3 エラー発見状況
Table 3 Status of the error detective

テスト対象	ウォークスルー (件)	単体機能テスト (件)	機能テスト以降 (件)	発見件数 (計)	発見率 (ステップ)
発注商品登録プログラム (オンライン)	20(18)	4	1	25	55
仮COS MR登録プログラム	12(11)	0	0	12	63
発注商品補正プログラム	13(9)	1	0	14	114
スルーテスト(バッチ系)	未実施	2	0	2	212

ウォークスルーで発見のカッコ内の件数は標準規定の記述ミス件数
発見率：プログラムのプロシージャステップ数/エラー発見件数

ローチの両形態が採用できるようにワークシート類を用意したり，また中工程では既存システムの改造から出発する場合のワークシート類を用意している等，ケースに応じて利用するワークシートを選択できるようにしている．また，そのシステムではどのワークシートを利用しているかを把握できるようにしている．

主なワークシート類をカテゴリ別に図12に示す．

4.2.5 アクティビティ管理

ある仕事を与えると，それなりに実行計画を立てて進めていくが，チェックしてみると大幅に遅れていたり，進め方がわからずとまどっているケースがある．

SEの場合はそのことが他の人を遊ばせてしまう原因にもなる．また，新たに採用した協力会社要員の場合（派遣要員）どの規模から仕事を与えていけばよいかとまどうことになる．

本来的には，より抽象的な規模の大きい仕事をまかせられることが望ましい．まず当人の力を計り，まかせる人とまかせられる人の抽象度合を大きくしていくために設けたのが，この管理方式である．これは，SIS (Short Interval Scheduling) と呼ばれる古典的な作業管理手法を改造したものである．

ある仕事をまかせる．その仕事をアクティビティ（細分化した仕事）単位に分解し，アクティビティノートに起票する(追加記入できるようにしておく)．そしてこれをベースに実行計画を立てる．このとき，アクティビティの単位をその作業のアウトプットとして規定したワークシート作成よりも小さく分解できるかどうかをチェックしてみる．それができれば，まずその仕事をまかせてもよいと判断する．そして次回からは，さらに大きな仕事の単位を与えていく．このときアクティビティノートへの起票は，ワークシート出力の仕事単位とほぼ同等の単位としている．第1回目でワークシート作成よりも小さく分解できなかった場合は，日次単位の仕事から与えていく．そして徐々に仕事の単位を大きくしていくわけである．

この方式はその人の気質にもよるが，新人に3か月後には，後工程（プロジェクト仕様作成～テスト計画・実施)の仕事をほぼまかせられるようになることと，SEとPEの要員を階層構造的に育成していけることに大きなメリットがある．

4.2.6 工数見積りと役割分担

システム開発規模算出は，図13に示す方式を採用している．

また，算定の基準として次のような考え方をとっている．

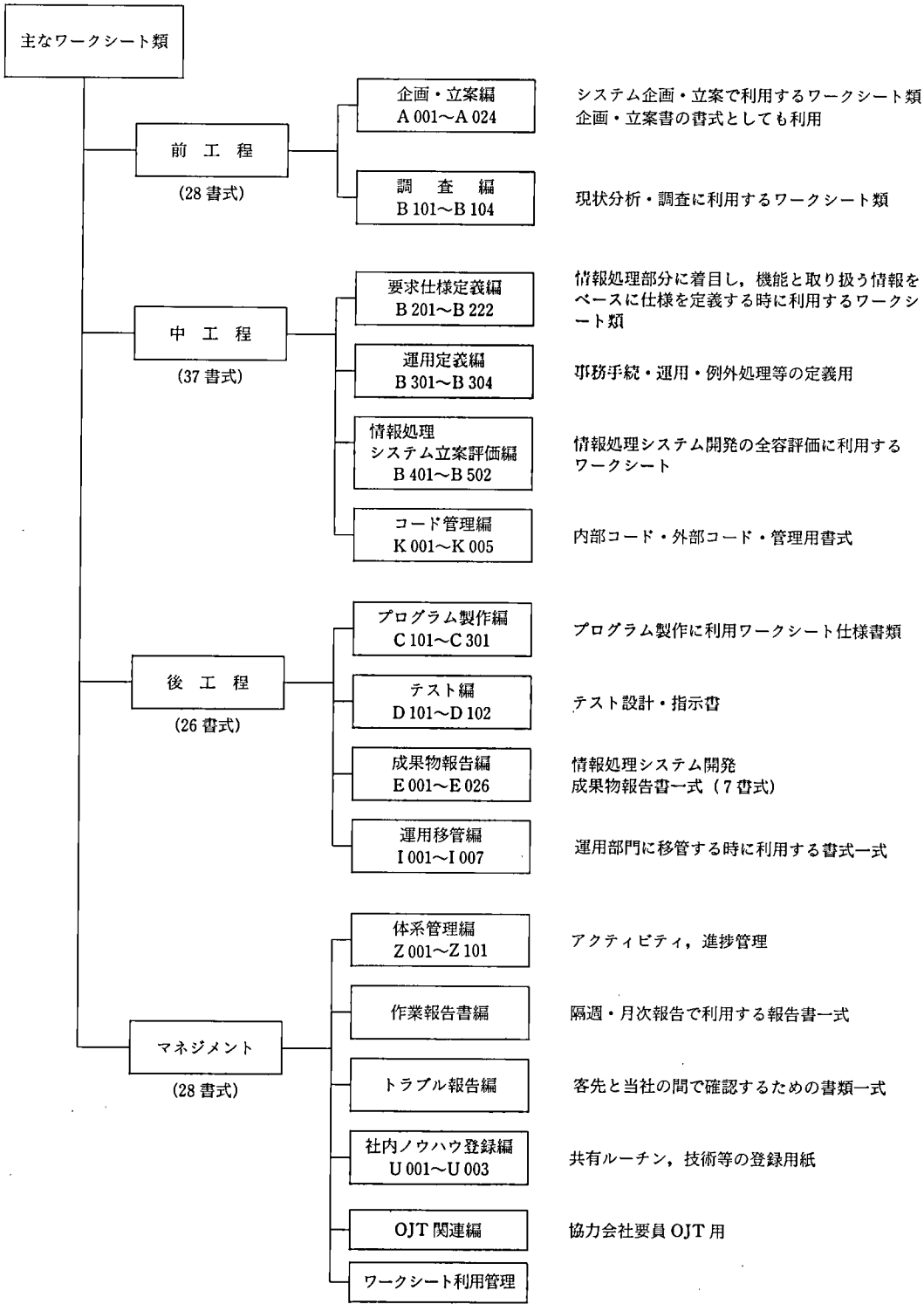


図 12　ワークシート類一覧

Fig. 12　Summary of worksheets

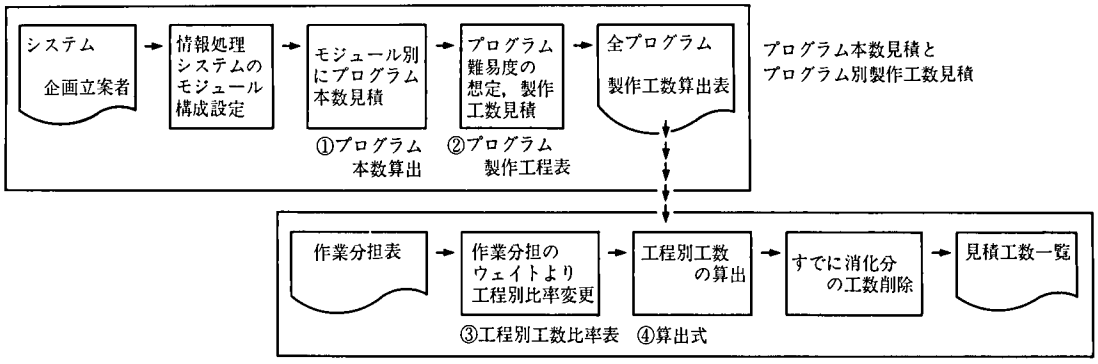


図 13　開発工数見積り算出方式

Fig. 13　Development manpower estimating

1)　この見積り算定の範囲は，情報処理システム立案～製作～稼働後支援までを含んでいる．ただし，EDP 外システム（教育・制度改訂・運用マニュアル作成等）は作業範囲に入れていない．

2)　この時点（情報システム立案終了時）での見積りは，「機能」「概念ファイル」「概念トランザクション」「仕事の仕組み」「情報処理システム範囲」を算定要素としている．

プログラム本数の算出と難易度づけは，次に示す考え方に基づいている．

1)　モジュールで取り扱う要素（入力・出力，内部ファイル・外部ファイル・照会等．これらは，概念トランザクション，概念データベース，機能より想定する）を洗い出しプログラム単位を決める．

2)　取り扱う要素と要素を構成する項目数から，プログラムの難易度とプログラム本数を決定する（ここでは，処理ステップ 400 を標準に本数を算出する）．

3)　難易度は取り扱う要素（主にファイル数）とデータ項目数から決定する．難易度は，単純（ファイル数 3 以下，項目数 50 以下），平均（ファイル数 4 以下，項目数 100 以下），複雑（それ以上）の 3 種類である．

算定の基準値は表 4，表 5 の値を利用している．

ここまでは標準見積り工数の算出であり，この値をベースに実際にこの作業をやり遂げるのに必要な見通し工数を算出する．

見通し工数算出には二つの観点がある．一つは，この作業を外部に委託する時に委託分の工数を算出する場合であり，もう一つはシステム開発環境等の条件により，変動する工数を算出する場合である．

この両ケースの算出には，工程ごとにあらかじめ定めた作業分担表の作業項目一覧を利用する．

客先と，あるいは委託先と役割り分担を明確にするわけである．さらに環境設定等のシステム開発に必要な仕事と工数を算出する．役割り分担により作業工数は変動(配分が変動するだけでなく，担当する人により総工数も変わってくる．とくに検収，テスト計画等で大きく変動する）し，新たな仕事の追加あるいは削除によっても変動する．これらの変動要因を加味し，見通し工数を算出する．

見積り基準値として採用している値・割合は，数年間にわたる作業実績値〔プログラム作成のみ：2,500～3,000ステップ/月（5～6本/月），プログラム製作工程：1,200～1,500ステップ/月（3～4本/月），詳細設計～本番稼働フォロー：650～1,000ステップ/月（1.6～2.5本/月）〕をベースに設定している．条件を設定し基準値と算出式から工数を算出していれば，累積実績値を反映させることにより，次にはさらに精度の高い算出が可能となる．したがって，実績値を管理しておくことが見積りには最も重要であるとの考え方をとっている．

4.2.7　利用者部門主体のシステム企画・立案工程

これは，別段目新しい事項ではなく従来からのやり方に都度工夫を加えてきているものである．

ベースは，当社のシステム開発技法RSDM（Reliable System/Software Development Method）の企画・立案工程の考え方と技法を採用しており，作業を具体的にするためにワークシート類を整備し，さらに上位層の意向を取り入れやすいようにしたり，現状分析に陥らないように既存の仕組みについては情報処理システム立案工程まで触れないようにしたり，どうするかが先行しがちなため「何をするか」の定義付けにウエイトを置いたり等の若干の工夫を加えたものである．対象とする領域の広さと質により簡便法も採用できるようにしている．

この工程でのアウトプットは，成すべきことと仕方を規定したものであるが，同時に上申書の素材となるため，あらかじめ上申事項を設定しその観点からまとめられるようにワークシートを用意している．

利用者部門を主体としているが，そのために新たに設定した事項はなく，情報・機能・概念データベース等について説明材料を用意した程度である．

管理系情報システムでは，当社SE 3名・協力会社SE 3名・客先情報システム部門2名・利用者部門7名で5か月・60人月を費やし，企画・立案書をまとめ上げてきて

表4　プログラム製作工数表(COBOLプロシージャ4,000ステップベース)

Table 4　Manpower of the programming　(単位：人日)

種　別	仕様書作成	プログラム作成	テスト	定　義	合　計
単　純	1.5	2.0	0.5	0.5	4.5
平　均	1.5	3.0	1.0	0.5	6.0
複　雑	2.5	4.0	1.0	0.5	8.0

表5　工程別工数比率

Table 5　Manpower's rate of the programming

工程別工数比率表(この工程区分は作業分担表の区分に従っている)

作業分担別　工程	情報システム企画・立案	外部要求仕様定義	改造計画	詳細設計	製作	EDP外システム構築	本番稼働準備	本番稼働後フォロー	稼働評価	マネジメント
工数比率	—	15%	—	18%	49%	—	4%	6%	—	8%
換算比率			—	21%	58%	—	5%	7%	—	9%

(全体の85%)

表4のプログラム製作工数

いる．これが現在，順次本番を迎えている管理系情報システムの外部要求仕様である．利用者部門のメンバは，その後，基準・制度の設定，運用設計，システムテスト設計等の作業をこなし現在に至っている．

4.2.8 そ の 他

1) アシスタント女性の採用……ファイル・ドキュメント類の管理，コピー，進捗管理資料作成，実績登録・工数管理，仕様書整理等のワープロ操作，テストデータ投入，データファイル項目定義，各種下調べ，マニュアル管理等に採用したのがこの領域である．作業領域も広がっており，現在は1名であるがSE 3名に1名の割合で十分見合うと考えている．

2) 協力会社との端末接続……機密保持，運用方式等，客先の状況に大きく依存するが，客先開発系コンピュータとの端末接続は生産性の向上に大きく寄与する．現在4社14台を接続中である．

3) ワンアワーレクチャ・タイム（One Hour Lecture Time）協力会社要員の技術向上を狙った週1回の自主教育講座であり，現在は座長も運営もすべて任せている．

5. お わ り に

外部要求仕様の利用部門による確定方式，SEの気質に依存する領域と最低限の品質・生産性の保障等，今後さらに改善を図っていかなければならない領域は多々ある．さらに，これらの工夫とは別に以下のような問題がある．

1) 技術の陳腐化……後工程の生産性を見ると，ある大きな塊の場合プログラム製作のみなら5,000～7,000ステップ/人月の実績値が出ている．これは，アプリケーションレベルプロトコルの規定を行い標準機能を整備し，パターン化を進める部品を用意したからである．このことは非常に有効な手段であるが，同一環境下でのプログラム開発であり，技術の陳腐化を招く恐れがある．新しいソフトウェアプロダクトの機能を利用し，生産性と品質の向上を図ることと，既存のツール類，技術・部品を利用することとの踏ん切りに悩まされている．

2) 実績値の把握……ここ数年，当社から外部への業務委託を増やしてきている．協力会社との分業化を図ることは有益であるが，委託先の実質工数が把握できなくなってきている．実績値がつかめないことにより，見積りの精度が停滞してしまうことになる．とくに委託領域を増やしてきており，より前工程での工夫・成果がどれだけ貢献度として表れてきているのか不明になりつつある．

一方，情報システム開発の規模はとどまることなく大きくなり，領域は広がり，複雑化してきている．技法を導入し，工夫を重ね，ツールを用意したことも次には当たり前のことになり，さらに高次元な品質・生産性が求められる．客先の要求に確実に答え，システムインテーグレータとしての役割をになうためにもさらに体系化された工夫を図っていきたい．

参考文献　[1] T. Demarco, 渡辺純一訳,「ソフトウェア開発プロジェクト技法」, 近代科学社, 1987.
[2] 藤野喜一・花田牧悦編集,「ソフトウェア生産技術」, 電子通信学会, 1985.
[3] G. J. Myers, (長尾真 監訳),「ソフトウェアテストの技法」, 近代科学社, 1980.
[4] 日経コンピュータ, 1987.9.14 号.

執筆者紹介　福　原　俊　作 (Shunsaku Fukuhara)
昭和 22 年生. 46 年神戸大学教育学部数学科卒業. 同年日本ユニシス(株)入社. 以来シリーズ 1100 ユーザのフィールド担当 SE として現在に至る. 58 年より小売業を主体とした流通ユーザを担当する. 関西支社システム一部所属.

K社新人事情報システムの開発

Development of Personnel Information System at K Company

若吉修治

要約 コンピュータ・システムの先駆けとして，大部分の企業において稼働しているシステムに，「人事・給与システム」がある．大量のデータを一括して処理するという，まさにコンピュータ向きの業務として，早くからコンピュータ化が行われ省力化の効果を上げている．

しかしながら，システム化の時期が早かったゆえに，種々の面で問題を抱えたシステムとなり再構築の対象としてよく取り上げられるようになった．

本稿は，「K社人事情報システム」の再構築の過程を述べるとともに，その開発に際して適用した開発技法，プロジェクト体制等に言及した．

Abstract The computer system which is now in operation at almost any corporation functions as a 'personnel/payroll system' that foreran computerized applications. The personnel/payroll system, known to be most applicable to computers because of their capability to execute batch processing of large-volume data, was computerized a long time ago and has continued to help save a great deal of manpower. However, the system has turned out to be possessed of problems in various aspects caused by the earliness of computerization, and has now been often taken up as a system to be re-constructed.

This paper describes the process of our re-construction efforts for K Company's personnel information system together with applied development techniques and the established project structure.

1. はじめに

近年，「人事情報システム」の再構築や新規開発が業種を問わず話題にのぼっており，その成果がコンピュータ雑誌に「特集」されているのを目にすることができる．

いわゆる「人事情報システム」には，給与に始まり，就業，税収，社会・労働保険，生保/損保，人事・労務統計，キャリア開発・教育訓練等，企業の社員およびその家族にまつわる，いっさいの業務が含まれている．このシステムは，企業が市場拡大のために社内の人材の実態を把握し，育成して人的資産を拡大するための「人材戦略システム」としての色彩を強めている．その充実の度合いは，企業が「社員を大切にしている」か否かの一種のバロメータといえる．企業にとって，これらの業務は膨大な量となっており，コンピュータ化が早くから手掛けられてきた分野でもある．

しかしながら，コンピュータ化の時期が早かっただけに，ハードウェア技術，ソフトウェア技術，システム開発技術等が未発達・未整備の段階で構築されたシステムが多く，構造面・技術面や，操作性・保守性等で種々の問題を抱えたシステムとなったものが多い．企業内のあらゆる分野でコンピュータ化が進んでいるが，過去に構築したシステムの見直し，再構築の動きも盛んである．「人事情報システム」の再構築は，前述の問題解消はもちろんのこと，企業を支える社員を単なる「人的資源」から「人材情報」としてとらえ直し，その有効活用を計るとともに，社員の福利・厚生・サービス・能力開発等の支援強化をめざしている．

筆者は，「K社新人事情報システム」の再構築プロジェクトに参加し開発に携わる機会を得た．今や，企業の人事情報システムは，旧来の給与計算を中心とした人事・給与システムとは比較にならないほど，広範囲かつ複雑・大規模なシステムになっている．このシステムの再構築プロジェクトは，開発の技術面やプロジェクト・マネジメントという面でも非常に興味深いものがあり，新規開発となんら変わらない点が多い．

本稿では，このシステムで使用した開発技法について述べるとともに，プロジェクトの特徴や評価について触れる．

2. K社新人事情報システム

2.1 開発の背景

K社における「人事情報システム」は，給与計算のバッチ処理（正確にはPCSによるカード・ベース・システム）からスタートし，企業規模の拡大，地方事業場の増加，海外事業への進出，業種/業態の範囲拡大に伴う人事諸制度の改変，各部門へのサービス支援強化等のために，順次機能拡張が行われてきた．しかしながら，従来システムの拡張では限界に達しており，次に示すような問題を解決できない状態となった．

1) 各サブシステムが縦割りでできており，関連のあるデータでもその都度，重複入力が必要である．
2) バッチ処理中心であり即時性に欠ける．
3) 外部団体とのデータ交換と，それに伴う事務量の増大に対応しきれない．

このような状況のもとで，K社では全社的な業務改革の一環として昭和62年2月に「人事事務効率化推進プロジェクト」を発足させた．

同プロジェクトは，人事・勤労事務の簡素化・効率化，人事情報の充実，厚生・サービスの強化・充実をシステム化の狙いとし，さらに

1) 本質に迫るイノベーションとなるよう，現状や過去の経緯，局部にとらわれることなく自由・大胆に発想すること，
2) 「サービスの向上」と「効率」を調和させること，
3) 人事部内のみならず，事業部制に対応した全社のインフォメーション・システムの一環として考えること，

の3点を命題にして「新人事情報システム」の開発に着手した．

2.2 システムの目的

開発に当たって，考慮した点を述べる．

効率化・省力化を図ることは重要であるがその結果，社員へのサービス低下を招くことのないようにしなければならない．社員が人事，福利厚生の諸制度，給与/賞与の明細，財形貯蓄残高，就業時間を知るために人事部門にいちいち問い合わせることは社員にとっても煩わしいことであり，人事部門でも初歩的な事項，プライベートな照会等に対応することは，業務効率の面から見て好ましいものではない．

このような点から，「サービスの向上」と「効率」の調和を意識したシステム化を行うとともに，効率化によって捻出した工数を，人事部門の使命である，社員の能力開発や福利厚生，サービス強化に振り向けることを目的の一つとした．

先のような点は，社員個人が直接自分で照会検索，申込等の入力ができるシステム

を用意することにより解決が可能である．また，社員の結婚や出産あるいは異動等により，身上異動が発生した場合等，届け出書類を何枚も書かなければならないことがある．このような手続も，システム化により１回ですむような解決方法が考えられる．

これらをまとめると，先に述べたシステム化の狙いを実現するために，「新人事情報システム」は，

① ビデオテックスシステム採用によるデータの発生点即時処理，
② 従業員専用端末による本人の直接入力・検索，
③ 個人異動データのサブシステム間連動処理，
④ 人事事務センタ集中化，

等を目的としている．

「新人事情報システム」の全体図を図１に示す．

2.3 システムの概要

新人事情報システムの主要機能を以下に紹介する．

1) 異動データによるサブシステム関連処理……従来のサブシステムでは，一人の異動データ（たとえば，入社，退社，結婚，出産，勤務地異動）が発生した場合，各サブシステムごとにデータの投入が必要になっていたが，新システムではサブ

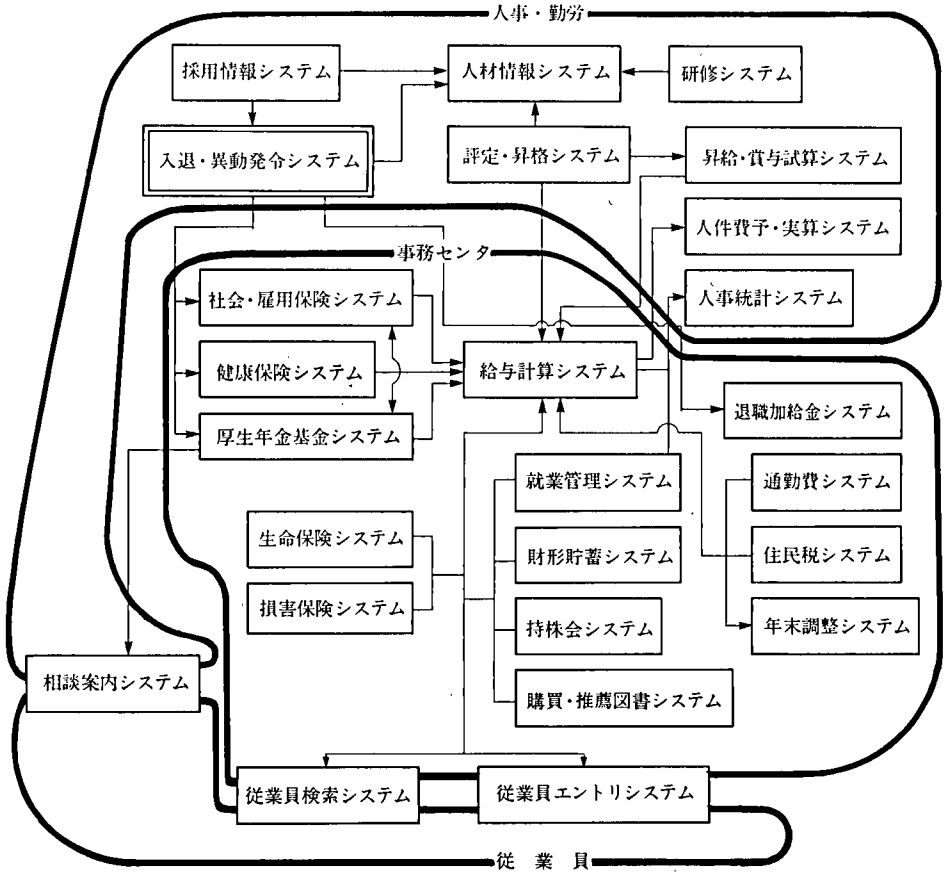


図 1 新人事情報システム体系図

Fig. 1 New personnel information systems

システム間でデータの受け渡しを行い，データ投入は１回ですむようになった．図２に人事業務の One-action 化を示す．

2)　社員用ビデオテックス端末の開発・設置……社員個人が，直接自分で就業データ（残業時間の申告，休暇等の申請）や，各種の届け出，申込みをビデオテック

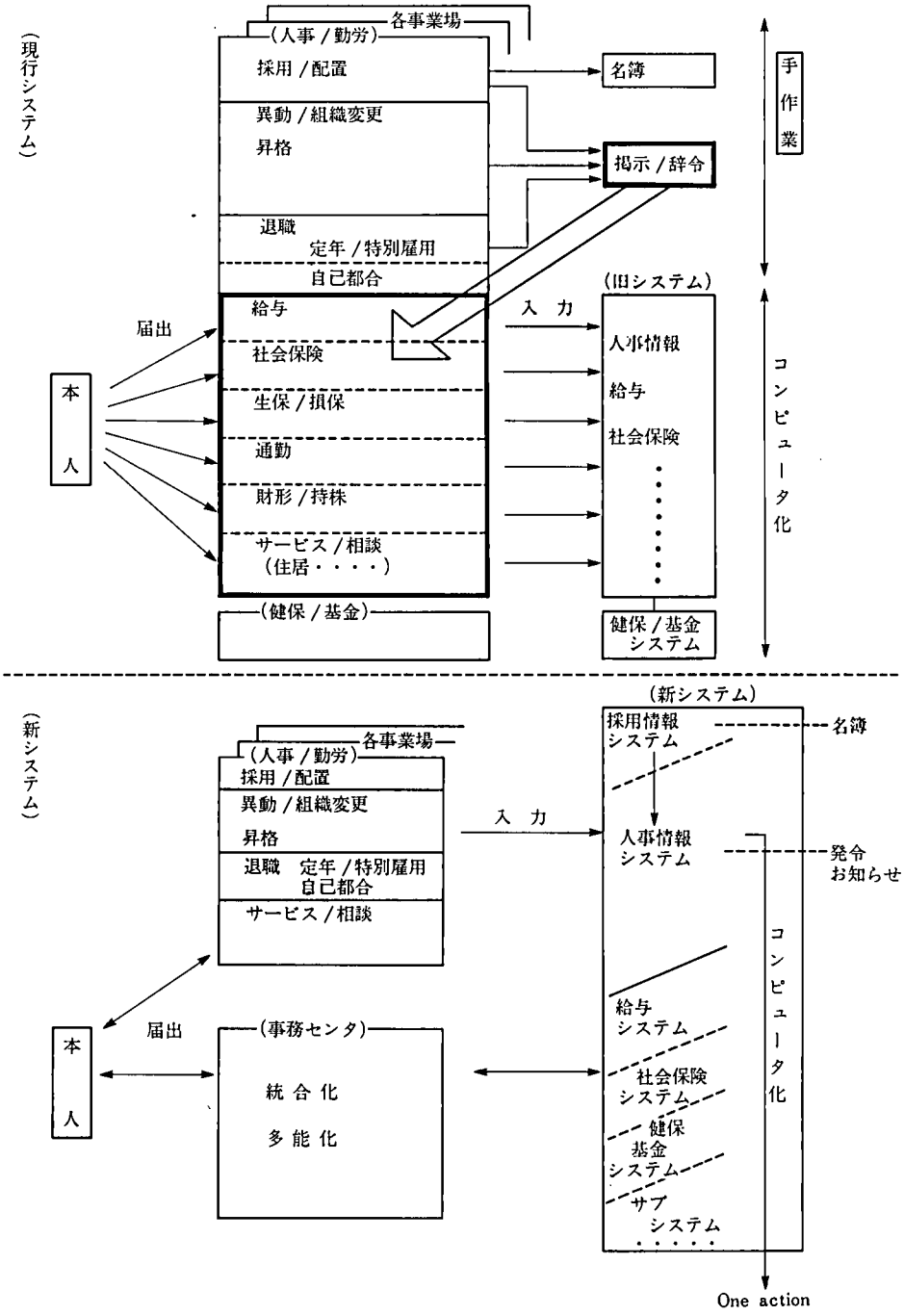


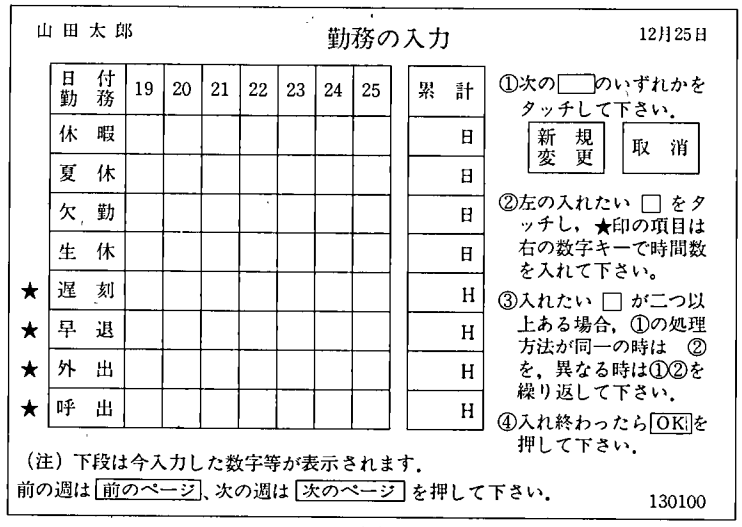
図 2　人事業務の One-action 化

Fig. 2　One action operation of personnel office work

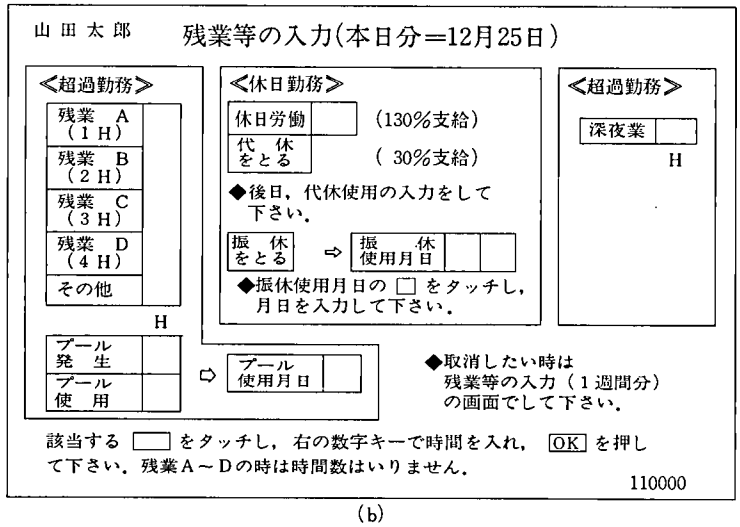
ス端末から行うようにした．図3にビデオテックス画面の例を示す．

3) 異動発令の自動作成……K社では，社員の入社・退社・異動は発令日当日に「人事発令掲示」という形で社内の掲示板に貼り出される．従来，この掲示の作成はワープロにより行われ，年間にすると本社・支社・事業場を合わせて約15,000枚にも達していた．新システムでは，異動の決定しだいデータ入力を行っておき，発令の前日に異動データとして処理すると共に，各部署の端末プリンタに掲示できる形で出力するようにした．

4) 社会保険等の届出作成……社会保険・労働保険関係のものとしては，従来届出書類は書式がそれぞれ異なることや，相手先が多岐にわたるため，すべて手書きで行われていたが 1)のサブシステム連動処理により，異動データをもとに必要な書類を作成するためのデータが自動生成できるようになった．



(a)



(b)

図 3 ビデオテックス画面の例

Fig. 3 Samples of VIDEOTEX screen

3. 開発技術

開発に先立ち要求機能をもとにプログラム本数を見積ったところ，23サブシステム600本（移行プログラムを含む）40万ステップ，開発工数500人月という膨大なものとなった．このような大規模システムのシステム開発プロジェクトは，多くのプロジェクトメンバを必要とし期間も長期間にわたることになる．

筆者は，このプロジェクトを成功に導くための要素として，次の3点を取り上げた．

1) 一定の開発技法が必須である．
2) 開発体制を整えることが大切である．
3) 問題の早期発見と対策のためのシステムが必要である．

3.1 開発技法

大規模開発において，プロジェクト管理の観点から注意しておくべきいくつかのポイントがある．一般には，品質(QUALITY)，コスト(COST)，納期(DELIVERLY)の3点といわれている．

ここでは，とくに品質の面に絞って述べる．品質を決める大きな要素は，プロジェクトの初期に行われる要求分析・要求定義の工程の成果の善し悪しである．それは第一に要求機能がすべて出尽くしているかどうかであり，次に要求機能が明確に定義されているかどうかである．そして，要求機能が正確に開発者に伝わっているか，また要求機能について相互の理解に誤りはないか，等である．これらのポイントはすべてのプロジェクトに共通の問題であり，早期に解決すべきものである．プロジェクトの初期段階にこの種の問題をクリアにしておくことが，開発を行っていく上で一番重要な点といえる．プロジェクト・マネジメントに関する文献でよく取り上げられている「不確定要因（Uncertainty)」がこれである．

通常「不確定要因」は時間とともに減少していくが，プロジェクトの成功はこの減少カーブをいかに早く下げるかにかかっているといっても過言ではない．

図4は，不確定要因が時間（工程）と共に減少するカーブを表したものである．

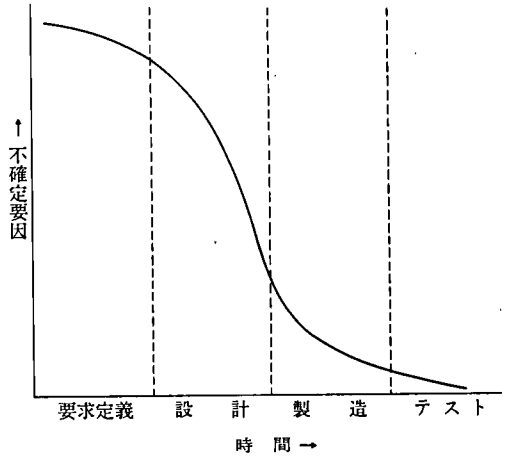


図 4　不確定要因の属性

Fig. 4　Attributes of uncertainty

筆者は，「不確定要因」を早期に減少させる手段として，長年の開発作業を通じて採用し効果を発揮してきた「構造化分析技法」の一つである「DFD (Data Flow Diagram)」による要求機能の定義を行い，プロジェクト・メンバ全員に徹底することを最初のステップとすることにした．

「DFD」の特徴として，①図形表示を用いる，②分割されている，③多次元的である，④データの流れを中心に書いている，⑤コントロールの流れに重点を置いてない，等があげられる．

これらのことはまた，次のように言い替えることができる．

1) 図形表示のため，誰にでもわかりやすい．

2) 一つ一つの機能が分割されていて，誤解が生じにくい．

3) データの流れが中心であり，理解するための情報が具体的に表現される．

4) EDP の知識のない人でも，書くことができる．

このような特徴を持つ「DFD」を，要求機能の定義に利用することにより業務経験レベルの多少，EDP 知識の深浅に関わらず多数のメンバ間の相互理解を助ける共通言語としての役割を果たすことができる．

「DFD」による要求仕様の定義を行うことにより，プロジェクト・メンバ（K 社，当社双方）が，システム・イメージを早く理解できる形にまとめることができた．

図 5 に DFD の例を示す．

3.2 開発の体制作り

大規模システム開発を実施するためには，多くの開発要員が必要となる．メンバの人数が増えるに従い，とくに考慮しておくべきことに，メンバ間の情報伝達をいかに円滑に行うかがある．

たとえば，

1) 多数のメンバが同じ目標に向かっていることを，どうやって確認するか，

2) 人から人へ伝えたいことが正しく伝わっているか，

3) 共通事項を全員が認識しているか，

等である．

情報のスムーズな伝達と，理解を早めるための手段が不可欠となる．また，メンバがそれぞれの役割を正確に知っておくことも大切である．

開発プロジェクトとしては，ごく当り前のことであるが以下の項目をプロジェクトとして実施した．

① メンバの役割を明確にする．

② 工程を分割し作業内容を定義する．

③ 工程ごとの成果物を決め次の工程の入力とする．

具体的には，4 章の開発体制の項で述べる．

3.3 問題の早期発見と対策

3.1 節でも述べたように，開発作業の過程ではさまざまな不確定要因が原因となる問題が発生する．たとえば，①要求が明確でなく機能定義ができない，②仕様に矛盾があり，手戻りとなる，③見積りと実績に大きな誤差があり納期が守れない等である．

このような問題は，プロジェクトの進行上避けて通れないものであるが，あらかじ

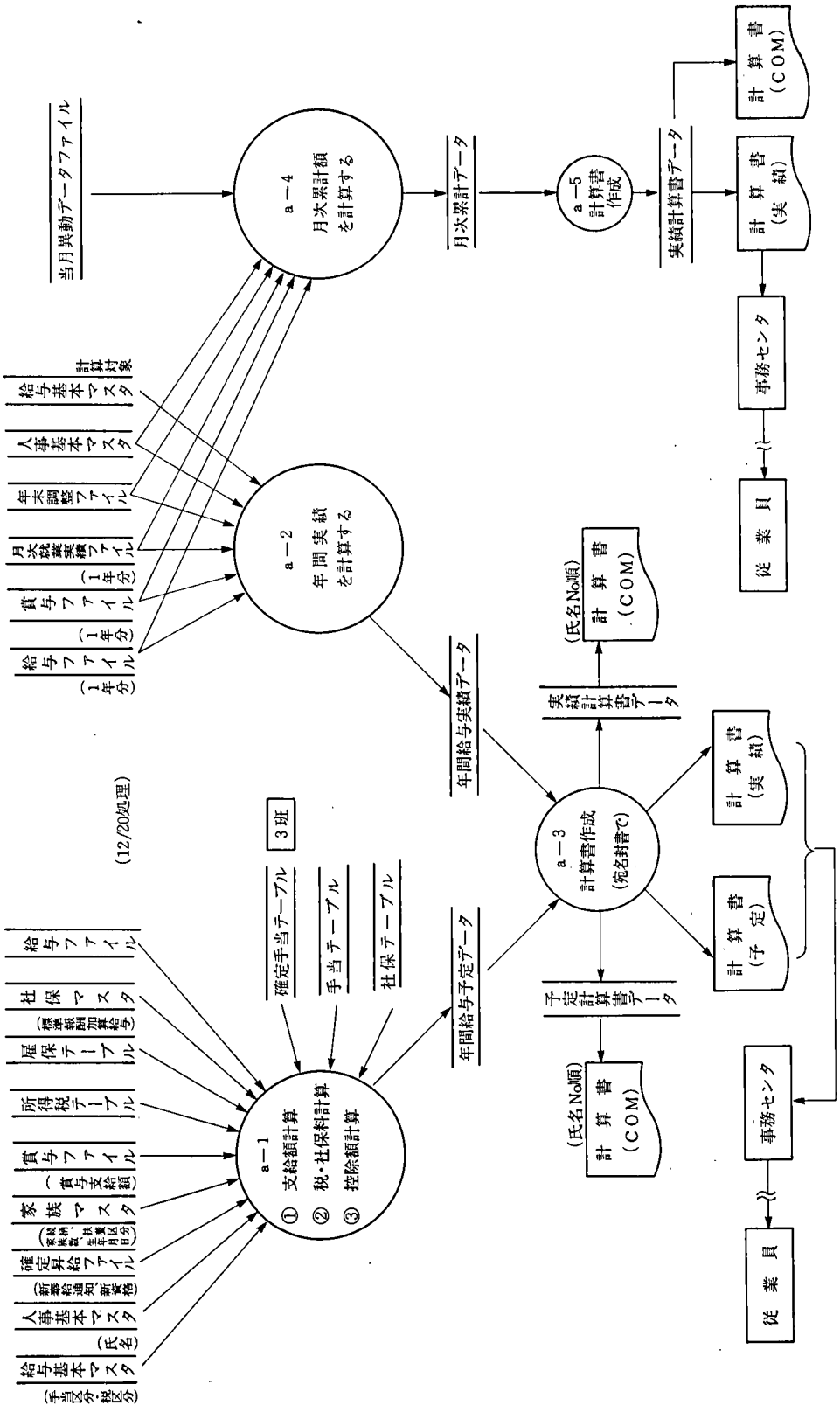


図 5　DFD による要求機能の定義例

Fig. 5　A sample of requirement specification by DFD

め予測して対策を取ることにより，未然に防ぐことができる．一つは3.1節に述べたごとく顧客の要求を明確に把握することであり，もう一つはプロジェクトとして問題の発生を早期に把握し対策を用意しておくことである．

その手段として筆者は「プロジェクト計画書」を作成し，そのレビュを通じて上記問題点の発見と対策をとった．「プロジェクト計画書」のレビュを，定期的に（3週間に1回程度），あるいは工程の替わり目に実施することにより，とくに向こう2～3か月以降の開発予定とその要員計画の変更/修正の可否，あるいは工程の成果物の品質等を的確に把握することができた．

たとえば，「プロジェクト計画書」のレビュにより発見され，対策をとったものとしては，

1) K社プロジェクト・メンバの結合テスト要員不足とその増員対策，
2) K社プロジェクト・メンバの「仕様書」の作成能力に合わせた要員配置の見直し，

等がある．

なお「プロジェクト計画書」とは，プロジェクトの発足時に以下の項目をまとめたものである．プロジェクトの初期の段階で考えられる問題をここに上げておきレビュすることにより，問題の発生を事前に察知できるようにするものである．

① 顧客の概要，製品，組織
② システムの概要，特徴，範囲，課題
③ システムの規模，ハードウェアおよびソフトウェアの構成
④ 開発の方針
⑤ プロジェクトの概要，特徴，体制，作業分担
⑥ 開発スケジュール
⑦ プロジェクトの課題

これらの項目は，プロジェクトの進行に伴い随時更新されてていく．

たとえば，今回の例では①～④についてはあまり変動はなかったが，⑤～⑦は内容が変化し，さらに

⑧ 開発場所，開発機器
⑨ 予算管理（開発予算の計画と実績の管理）
⑩ 教育訓練
⑪ テストの体制，移行計画
⑫ 運用体制，運用手順
⑬ 保守体制，引継ぎの問題

等の項目が追加されていった．プロジェクトとしての問題をいかに明確にしていくか，明確にでき得るかがこの「プロジェクト計画書」を作成する上でのカギとなる．

4. 開 発 体 制

4.1 K社の開発体制

開発に当たり，K社より当社に開発の応援依頼があり，作業がスタートした．K社のプロジェクト・メンバは，以下の4グループおよび全体総括1名の計11名で構成さ

れていた.

1)　就業・給与事務効率化グループ(3名)

2)　厚生・サービス事務効率化グループ(3名)

3)　パーソナルデータ活用グループ(2名)

4)　開発基盤/運用グループ(2名)

メンバ中,システム開発経験者は4)グループの2名と1)～3)のグループの8名中3名であり,他のメンバは実務部門の出身者であった.EDP未経験者が多く含まれている点が特徴といえる.また4)グループはシステム開発部門,他のメンバは1名を除き人事部門の所属であった.このプロジェクト・メンバによりシステムの要求分析が進められ,当社に依頼があった時点では,プロジェクトとしての開発要求アイテムがほぼ絞られ,「人事事務効率化推進プロジェクト報告書」という約200ページ強のドキュメントとしてまとめられていた状態である.

4.2　当社の開発体制

当社の開発メンバの体制および役割は次の通りである.

1)　ベーシック担当(開発基盤の設計および構築支援　3名)

2)　アプリケーション担当(業務プログラムの設計および開発支援　5名)

アプリケーション担当のメンバは要求定義がすでに終わり,それに基づいてEDP機能の定義をK社プロジェクト・メンバが行うことが決定していたため,ベーシック担当の基盤整備の作業と並行してEDP化要求機能の定義を行うドキュメントである「プログラム機能仕様書」の作成支援を第一の目的とした.

アプリケーション担当の役割を次にあげる.

①　「プログラム機能仕様書」の作成支援
　□プロセス設計
　□「プログラム機能仕様書」のレビュ
　□プログラムの発注

②　品質管理
　□各種標準/手順の作成
　□標準/手順の教育,指導および徹底

③　プログラム製造担当者に対する技術支援

④　進捗管理

⑤　単体検収作業

⑥　結合テストの支援

⑦　グループ間調整

4.3　開発工程と役割分担

要求定義工程以降については,図6に示すように定義し実施した.仕様書以降については,作業項目を細かく定義し個々の作業の途中,あるいは最後でアプリケーション担当がレビュを行うことにした.この方法により,①工程ごとの成果物の品質を保つ,②要求機能がEDP機能に反映されているかをチェックする,③互いの役割を相互に認識できる作業に専念できる,等の効果があった.

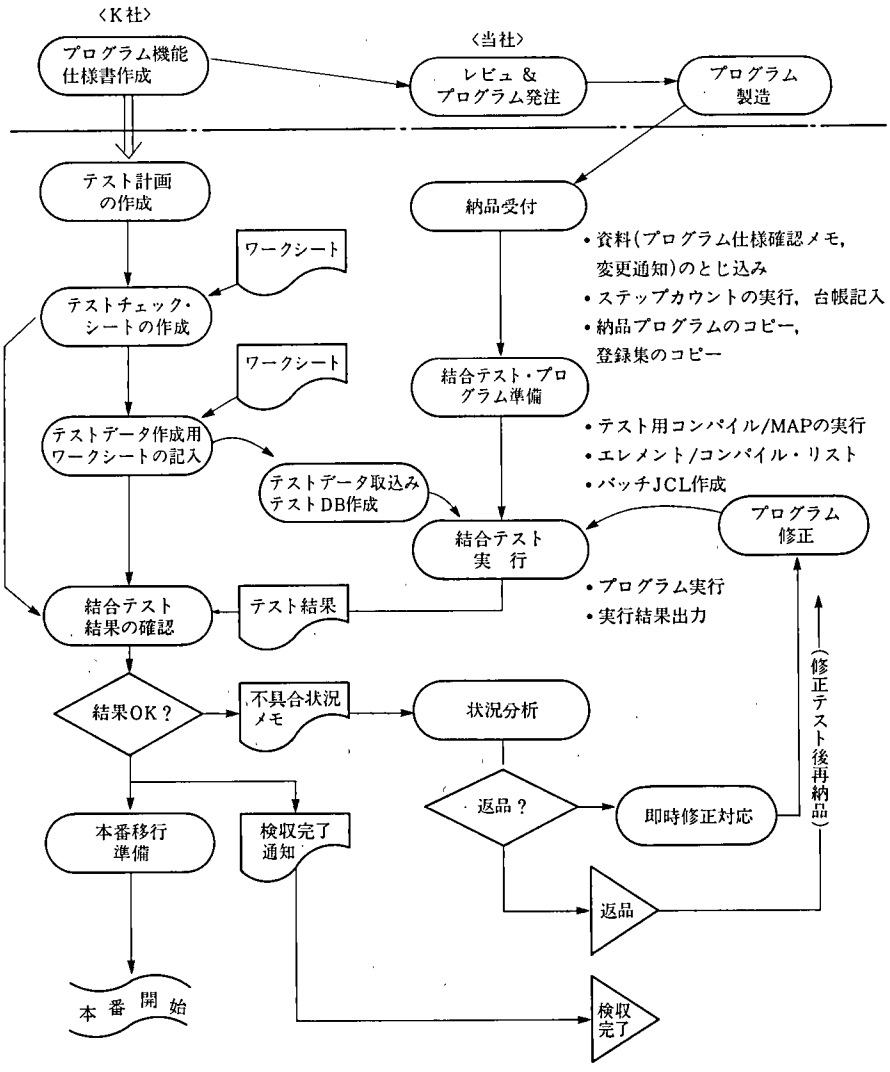


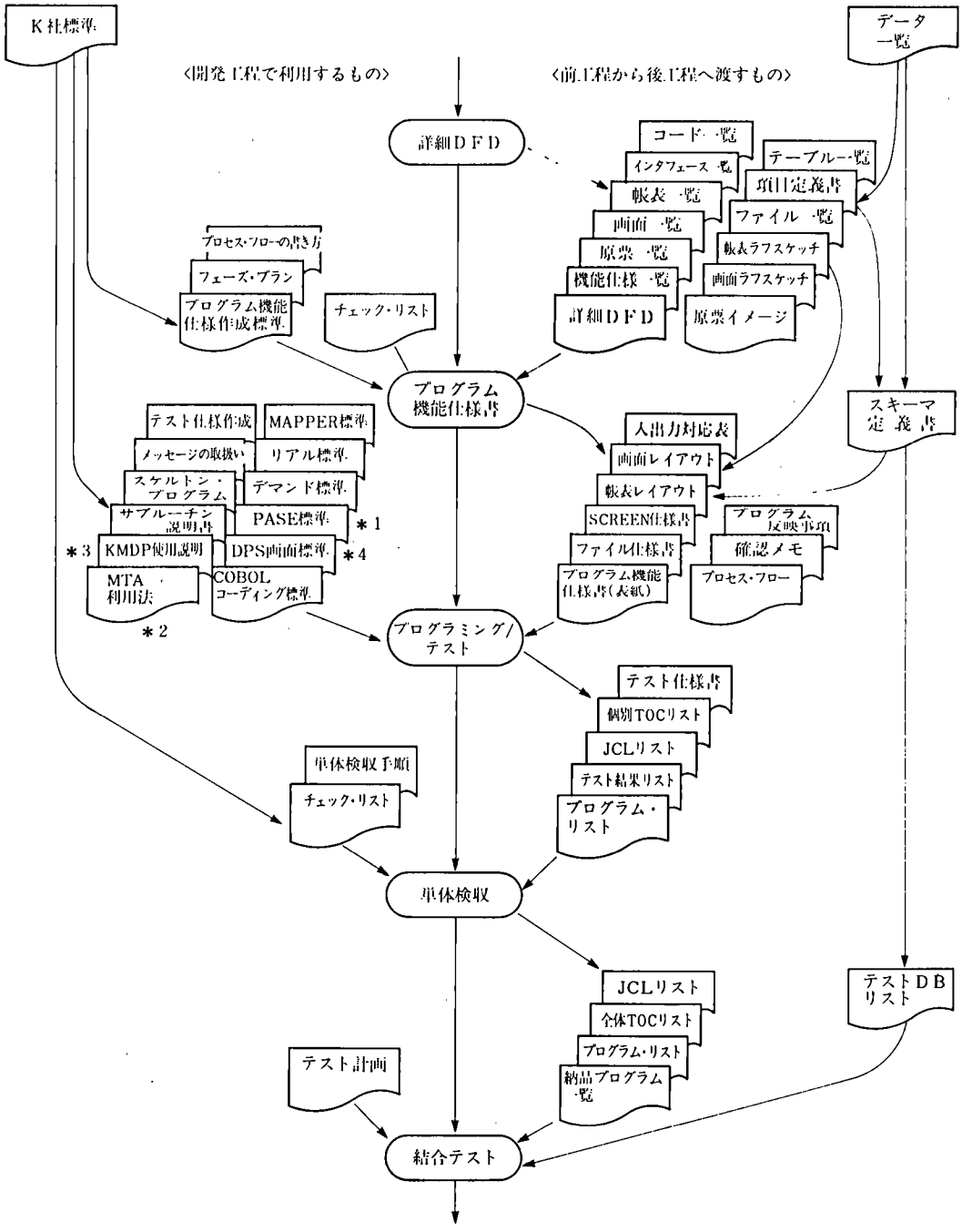
図 6 開発工程と役割分担

Fig. 6 Development process and job assignment

4.4 開発工程とドキュメント体系

EDP 未経験者が，半数以上であるため要求が正確に伝わるように，図 7 に示すような工程とドキュメント体系を決めて実施した．K 社担当者の意図する要求機能がもれなく下流工程に伝わり無用な誤解や勘違い，聞き漏らし等を起こさないような方法であると考えた．

通常，人は時間とともに記憶が薄れていくものであり，このような問題を避ける意味でも工程から工程への情報伝達は，必ずドキュメントを義務付けた．書き物にすることは，手間が掛かるという欠点もあるが，一方で物事を整理して書くという習慣や，愛昧な記述ではおのずと疑問が沸き，問い正すというアクションにつながることで効果を上げることができた．紙に残ることで記憶が薄れる，忘れるということもなくな



＊1 PASE：UNISYS1100/2200シリーズの仕様記述言語パッケージ
＊2 MTA　：UNISYS1100/2200シリーズのモジュール・テスト・パッケージ
＊3 KMDP：K社独自のデータベース・インタフェース・ルーチン
＊4 DPS　：UNISYS1100/2200シリーズの画面処理パッケージ

図 7　開発工程とドキュメント体系

Fig. 7　Development process and document system

る．さらに，保守フェーズになったとき，保守担当者が最初のシステム担当者の意図を知る手掛りにもなり，保守する上で非常に有効な資料になりうる．

5. 評価とまとめ

筆者にとって，総開発量600本40万ステップのシステムの開発中心メンバとして携わったことは，たびたび経験できることではない．本システムの開発中，その過程を通じて得たことを以下にまとめてみる．

5.1 評　　価

要求分析が終了し，開発要求アイテムが出尽くした状態で開発メンバとして参加するはずであったが，実際の開発過程では開発要求アイテムが相互に矛盾を持っていたり，分析のレベルに差があり，要求分析に戻ってやり直すという部分もまれにあった．しかし，この種の問題はどんなプロジェクトでも起こる可能性はある．いかにそれを早く摑み，適切な対策を取るかがプロジェクト管理のポイントだと考える．とくにプロジェクトの初期の段階にこの種の問題は多く発生しやすい．

本プロジェクトでは，「構造化分析技法」の一つである「DFD」による表現を，要求定義の記述過程で実践した．EDP要求機能の表現方法は，全プロジェクト・メンバに理解され，比較的早めに「不確定要因」を減らすことができたらと考えている．その後「DFD」は，K社の他システムでの開発においても利用されるようになり，次第に定着/活用されるようになったものと考えている．

また，この技法の効果という面では，このプロジェクトは図8の点線の状態にあったと考えている．

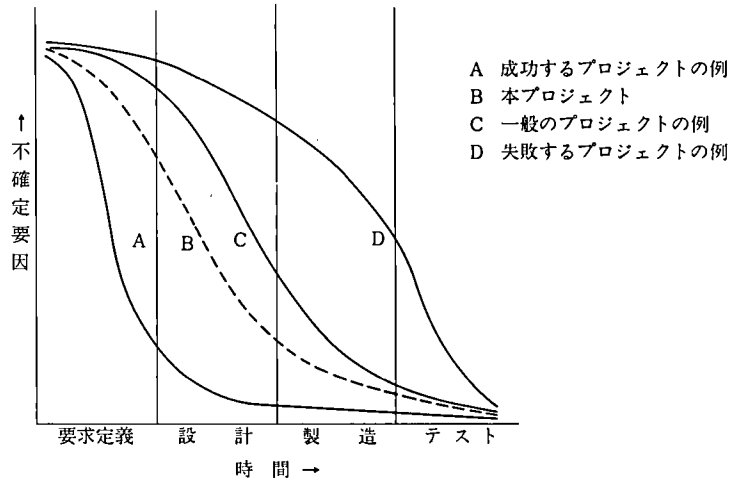


図 8　不確定要因の変化

Fig. 8　A variation of uncertainty

5.2 ま　と　め

本システムは，昭和62年5月の開発プロジェクト発足から，63年12月のプロジェクト終了まで，1年8か月にわたった長期プロジェクトであった．開発されたプログラム(移行プログラム，開発用ツールを含む)はCOBOL/MAPPERを合わせて約600

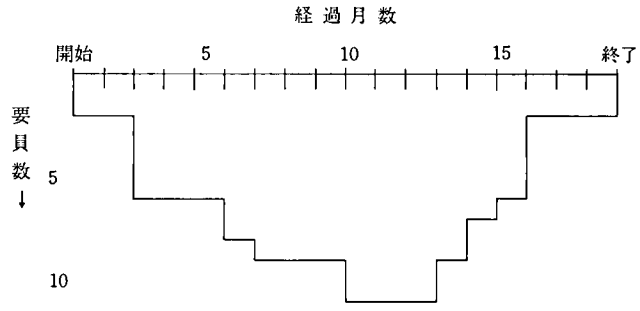


図 9　SE 投下工数
Fig. 9　Man hours of SE

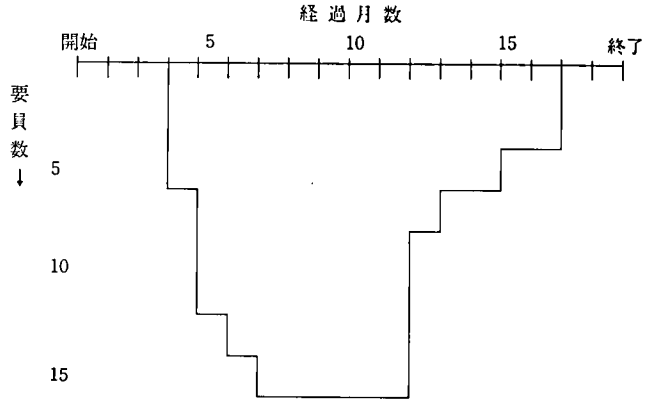


図 10　プログラム製造投下工数
Fig. 10　Man hours of programming

本 40 万ステップ弱であり，ほぼ見積りと一致した．その間，当社開発メンバは，2 名から最盛期 11 名（プログラム製造工程を除く）まで大きく変動した．

図 9 および図 10 に開発期間中の要員数を示す．

6. おわりに

「K 社新人事情報システム」は，昭和 63 年 12 月にプロジェクトが解散となり，保守のフェーズに入った．システム化の効果は徐々に発揮されつつあり，当初の目標を達成できたものと思われる．

本プロジェクトで試みた技法は，すでに当社の他のプロジェクトでも実践されつつあり効果が確かめられている．「システム開発に近道はない」以上，着実に効果の上がるやり方を地道に実践することが一番の近道ではないだろうか．本稿で述べた内容が，他のプロジェクト担当者の方々の興味を引き，いくばくかの参考になれば幸いである．

参考文献　[1] 矢野　功，「事務効率化とサービスの向上をめざす新人事システム」，ユニバック研究会全国会議第 25 回大会入選論文．
[2] 坂牧　勉，横田長次郎，小西徳雄，「K 社における VIDEOTEX を応用した人事情報システム」，昭和 63 年度テクニカル・シンポジウム入選論文．

［3］ T. DeMarco,「構造化分析とシステム仕様」, 日経マグロウヒル社.
［4］ 日本ユニシス,「戦略的情報システムの構築—開発プロジェクトの運営からシステムの評価まで」, 第15回ユニシスATS海外セミナー渡米研修団報告書.

執筆者紹介 若 吉 修 治 (Shuji Wakayoshi)

昭和46年大阪府立大学経済学部経営学科卒業. 同年日本ユニシス(株)入社. 官公庁, 流通業のSEサービスに従事した後, 製造業のアプリケーションシステムの設計・開発に携わる. 現在日本ユニシス・ソフトウェア(株)製造流通統括部生産システム一部所属.

多元的組織環境でのシステム開発におけるマネジメント

Systems Development Management in the Multi-dimensional Organization Environment

中 村 祥次郎

要 約 最近の大規模なシステム開発におけるさまざまな混乱（納期遅延・生産過程での工数の爆発的膨張等）は，商用コンピュータが普及し始めた頃に得た知識，経験がそのままでは通用しなくなっていることを物語っており，現実の動きに追い付けない事実にマネジメントが気付いていないことに起因している．つまり，新しい知識，新しいアプローチ，新しい視点での考え方が必要となってきたのである．言い替えれば，マネジメントの技術革新とも言うべきものを，社会が今求めているのである．

今日，われわれをとりまく職場（システム開発現場）は，次の点で昔のそれと異なる．第一に，ハードウェアの技術革新によってコンピュータ化可能な分野が広がり，一つのサブシステムに留まらず他のサブシステムとも密接に関係付けられた，大規模で複雑なトータル・システムが要求されており，旧来とは質・量とも格段に大きくなってきている．

第二に，仕事を遂行する上で係わりを持つ他の組織とのチャネルが非常に多くなってきている．多くの組織と同時に関係を持ちつつ仕事を遂行しなければならない高度で複雑な多元的組織社会となった．

第三に，知識労働者からなる組織体であり，これら個人・集団は論理立った概念や価値観によって動機付けられ働くのであって，高圧的な命令・指示では生産的な労働には結び付かない．

すなわち，知識労働集団から構成される多元社会では，その活動を生産的たらしめる高度で革新的なマネジメントが要求され，この課題の追求こそ組織の目的である顧客のニーズを満足させることにつながるのである．

Abstract Various types of confusion as seen in recent large-scale systems development (such as delayed delivery, explosive increase in production process, etc.), leading the current inapplicability of the knowledge and experience gained when commercial computers began to come into wide use, is attributed to the fact that management is still unaware of their inability to keep up with recent actual moves. In briek, a new knowledge, new approaches and fresh thoughs based on new viewpoints have come to be needed. In other words what is now sought is what is called technical management innovation.

Our working sites today - especially the sites for systems development - differ from old ones in the following aspects:

1) Continued expansion of computerizable application areas caused by technological innovations in hardware has spurred user needs for large-scale, complicated total systems which are not restricted to single subsystems but closely linked to other subsystems, and further their scale has been getting far larger and larger then ever both in quantity and quality.

2) Channels have become numerous in number through which we are concerned with other organizations in executing jobs. The multi-dimensional organization community, where we now ought to

communicate with a large number of other organizations while working, has also become more advanced and complex.

3) The organization we are in is made up of in tellectual workers. They, individuals or groups, are motivated to work by logically established concepts or by their own sense of value, and no high-handed orders or directions ever result in their productive working.

Conclusively, in the multi-dimensional organization consisting of intellectual workers' groups, all it takes to make their activities highly productive is really advanced and innovative management. It is only our efforts to follow this pursuit that enable us to find the way to satisfying customer needs, that is, the objective of an organization.

1. はじめに

われわれコンピュータ・システムの開発に携わる人間を取り巻く環境は，商用コンピュータが普及し始めてから約15年の間に激しく変化した．コンピュータの性能は，汎用機・PCを問わず50倍から100倍と大幅に向上してきており，システム規模も大規模化の一途である．今日に至っては，システム同士が密接に関係付けられた一大ネットワークを形成するまでになった．またコンピュータ化される分野が広がると同時に高度化されてきており，一開発担当組織ではシステムの機能から運用に至るまでの意思決定はむずかしく組織間の調整はもとより，組織そのものの見直しが必要になることも少なくない（図1）．

客先およびメーカの開発担当部隊としては，多くの他組織とチャネルを持たざるを

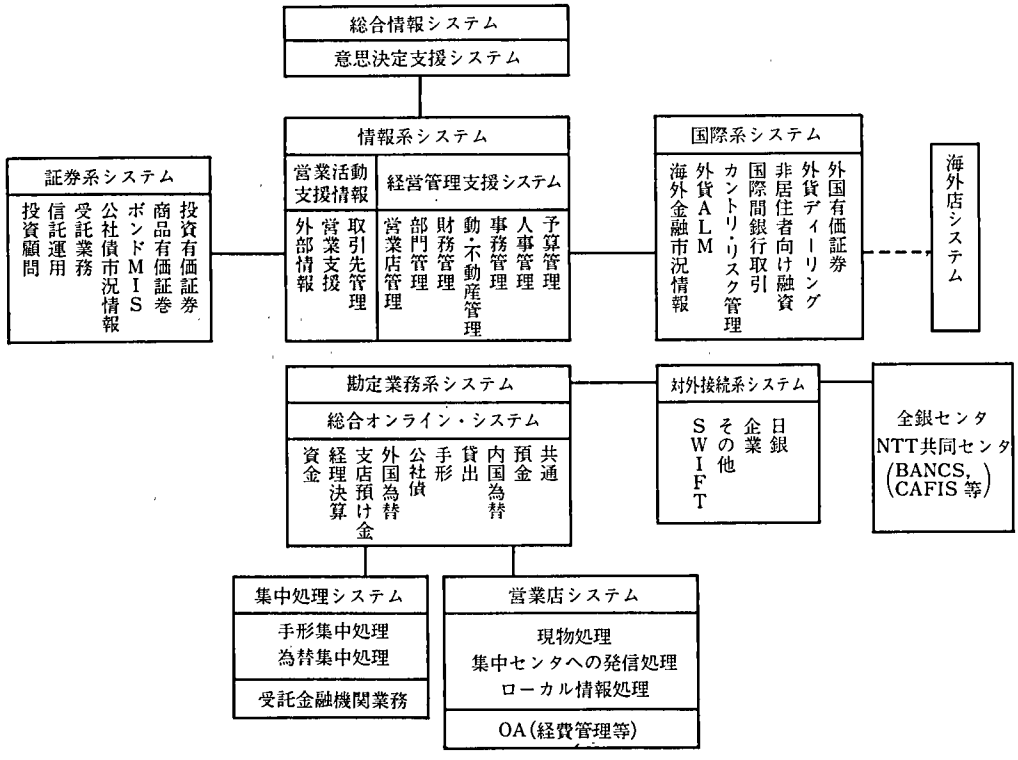


図1 システム規模

Fig. 1 System scale

得ず，多様な角度から折衝を行うことになる．このコミュニケーションを通じ，顧客の真のニーズを明確にし，最終成果物へ継ぐさまざまな意思決定を行うことになる．

このような状況に耐えうる組織とはいったいどのようなものなのか，また組織の活動を生産的なものとするマネジメントとは何か，基本的な考え方について考察を加え本稿にまとめた．

2. 環境変化

2.1 コンピュータ業務の拡大

コンピュータ技術は，ここ15年間に目覚ましい発展を遂げ，その適用分野も急速に拡大してきた．従来は人手に頼るものの，業務としては確立している分野の部分部分をコンピュータ化するに留まっていた．それが確立した全分野が対象となり総合システム化した．今日においてはコンピュータでないと不可能な分野や，業務としてその手続・運用が確立していない所までがコンピュータ化の対象範囲となり，コンピュータを前提とした新市場・新商品が登場するまでに至った．コンピュータの技術革新が従来なかった新たな市場を生み出したのである．

この新しい市場・ニーズに応えるには，新たな知識，新たなアプローチ，新たな視点が必要であり，新たな知識とは，AI，次世代言語，先物・オプション等の業務知識，オブジェクト指向的設計手法，インテリジェント端末としての端末ソフト等であり，新たなアプローチとしては，これら知識・技術を前提とした自社の商品にとらわれないシステムの構築である．

つまり，ユーザの求める構想を具現化し，自社の商品を当てはめ，足りない部分は他に求めるようなアプローチや多元組織を前提としたアプリケーション開発上のマネジメントが必要である．

新たな視点とは，ユーザ・ニーズを掘り起こし，具現化により，ユーザに満足をもたらすことを第一とし，誰が正しいかではなく，何が正しいかを大事にすることである．

最近の大規模システム開発におけるさまざまな問題（内工要員不足・納期遅延・生産過程での工数の爆発的膨張）は，従前で述べた環境変化に対し，古い知識・経験によるアプローチが，現実の動きに追い付けず通用しなくなってきていることを物語っているのである．

2.2 多元的組織集団の出現

コンピュータの技術革新により，コンピュータ化可能な分野が広がり，システムは大規模化の一途をたどる一方で，ますます複雑で高度なものとなってきたことは，初めに述べた通りである．

コンピュータの適用分野が広がるにつれ，組織間の調整・折衝はますます広がりを見せ，一開発担当組織ではシステムの機能から運用に至るまでの意思決定はむずかしく，客先およびメーカの開発担当部隊としては多くの他の組織とチャネルを持たざるを得ないため，さまざまな角度で折衝を行うことになる．

一方，システム規模が巨大化するにつれ，開発部隊そのものもいくつかの組織の集合体により形成されることになる(図2)．つまり各々立場の異なる組織の集団により，

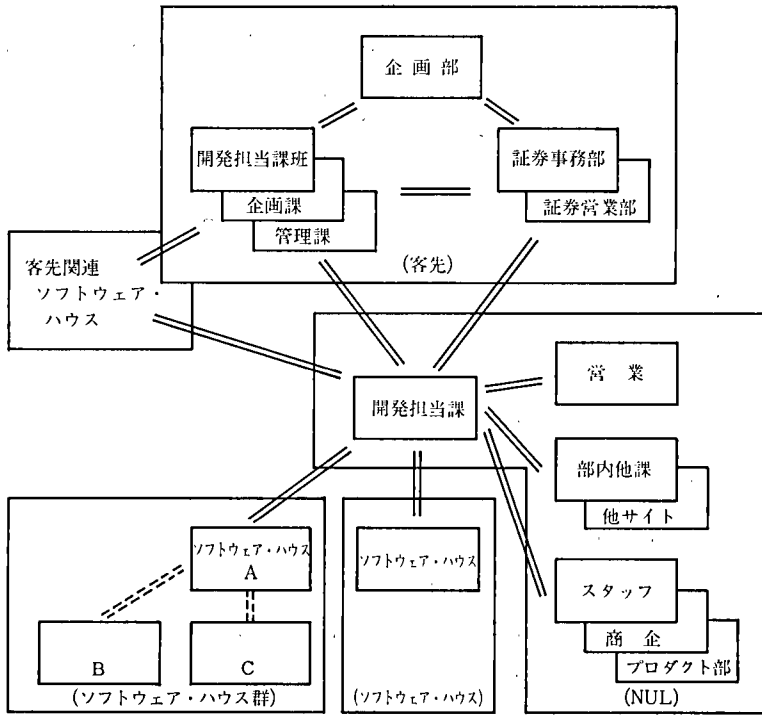


図2　多元的組織集団

Fig. 2　Pluralistic organization

システム開発が推進されることになる．この多元的組織集団の出現が，巨大で複雑なシステム以上にシステム開発を困難にしているのである．

したがって，この組織の集まりが各々の役割において責任を持って能動的に自立した一つの組織として，目的を達成することが最も重要であり，この組織の集団に目的を達成させるのがマネジメントであり，マネジメントの力である．

時代は，このマネジメント力を求めているのである．

3. 組　　織

3.1 組織の目的

組織は顧客の求める物を生産し提供するために，あるいは顧客の求めるサービスを提供するために存在する．したがって顧客の要求に応じられなくなったとき，その存在意義を失う．存在意義を失った組織は廃止されるべきである．一般的に組織集団において生産性が低下してきたとき，その組織集団の一部の組織が存在意義を失っている．つまり組織の目的は顧客に満足を与えることであり，そのための手段として存在する．しかしながら組織の活動が経済活動である限り，組織の活動を通じて利益を生み出さなければならない．この利益は，顧客の求める最終成果物を生産・提供するための条件の一つとして捉えるべきであって，組織の目的と捉えるべきではない．

P・F・ドラッガー著書「マネージメント」[1]の中で，『利益は顧客の満足に対するリスクをヘッジするものとして捉える』ことを勧めている．たとえば顧客のシステム開

発においては，システムの提案・見積り・設計・開発の各フェーズを通して，顧客の要求・満足に応えるべきであって最終成果物にのみ目を向けるべきではない．また，リスクの捉え方については各フェーズにより異なる．たとえば，見積りに対しては見積誤り，設計においては仕様変更，開発においては狙い通りの生産性が得られないという具合であり，各々の工程により異なる．このリスク・ヘッジとしての利益は，顧客のニーズに応えるための条件であって，それを要求することは健全な行為である．要するに顧客の理解が大事なのである．

3.2 知的労働者集団としての組織

知的労働に従事する知識労働者は，論理立った概念や価値観によって動機付けされ働くのであって，高圧的な命令や指示では生産的な活動には結び付かない．また彼等の活動現場においては，仕事の性質上その手順や基準をこと細かく定義することはできない．仮りに設定できたとしても，それは技術的に最低水準の決めであり，技能の向上と共に創意工夫の機会を失い生産性を低下させる．しかし自由であるがゆえにあらゆる判断が活動の中に織り込まれることになり，時には重大な問題についても平気で黙殺されることがある．たとえば，やる・やらない，続ける・中止する等の判断，つまり組織としての活動を正常に保つためには最小限の決めと指導・方向付けが重要となる．管理することにより組織の正常な活動を保とうとしても無駄な努力である．

また知的労働においては，労働時間は何の意味も持たない．技能や知識は道具でしかなく，知的活動によって出力された成果こそが重要となる．したがって，技能・知識を成果に有効に結び付けるために最大限の権限を委譲し，結果に責任を持たせるべきである．さらに言うならば，委譲すべき権限はその仕事に対する管理者の持つ権限すべてが対象となるが，委譲の限界は委譲される側の能力により決定される．また，時間の使い方は知的労働者の権利であり，成果物の生産に有効であれば良いのである．

一方，Dマクレガーは著書「企業の人間的側面」[2]で『人は仕事に対し欲求を持ち，達成と責任を欲する一面を持ち備えている』と言っており，知的労働はとくに当人のやる気に支えられている．やる気とは，向上心・達成欲・他人に認められたい欲求と不安である．また，知的労働に従事する集団は真のリーダを求めており，リーダに心理的に支配されることにより，もたらされる安心を求めているのである．やる気およびリーダの心理的支配の重要性は文献をあさるまでもなく，戦後の日本の経済的復興，その過程での日本と米国の関係が物語っている．

3.3 自立した組織

自立した組織*とは，組織の課題に対し責任を持っている組織のことである．自立した良い組織か，否かは，その組織が出力した成果で判断されるべきであり，組織を構成するメンバの技能・知識・経験とは無関係である．また，自立した良い組織は内に失敗や間違いを持っていても，マネジメント力でこれを克服し課題達成に向けて活動している組織である．内に何ら問題を持たずリスクをともなう仕事を拒否する組織は要注意である．信用すべきでない．

リスキーな課題に対し果敢に挑戦し，失敗や間違いをマネジメント努力で克服し責任を全うしようと活動する組織こそ真の自立した組織と言えるのである．また，組織

* 自立した組織：ここでは客先から見たNUL，またはNULと一括契約関係にあるソフトウェア・ハウスを言う．

を相手に仕事をする場合，相手の痛みを知らねばならない．腰を低くしなければならない．要するに気配りが大切なのであって，気配りのゆき届いた所では心地好く仕事ができるのである．自立した組織と組織が良き協力関係を維持し，強調し合う環境があれば，そこでの活動は生産的なものとなる．

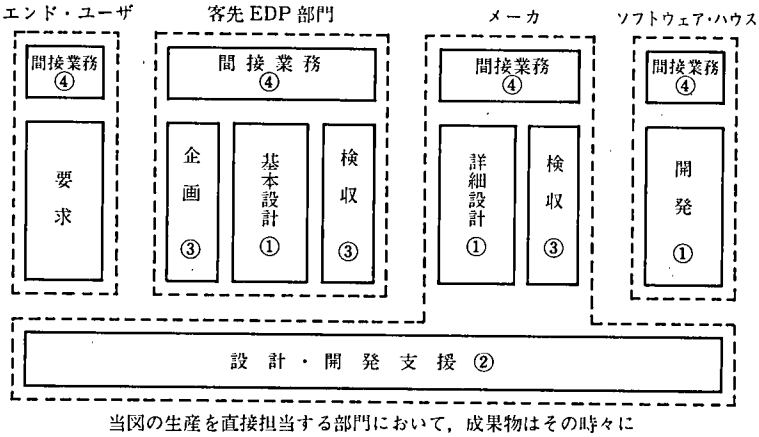
自立した組織を指導する場合，指導する側は誠実でなくてはならない．知識・技能のレベルが低いことについては，人・組織は寛大であるが不誠実なる行為には，ついて行けないのである．指導すべき組織・人間の欠点に目を向けるのではなく，組織の出力である所の成果物・生産性に目を向けるべきである．現象に目を向けるのではなく，真の原因に目を向けるべきである．問題の指摘，解決に向けて直接行動は慎むべきであり，マネジメントを指導するのが望ましい．内政干渉は自立した組織から責任感を奪い，依頼心の強い組織へと堕落させてしまうだけである．

3.4 組織の構成と規模

大規模システム開発を実施するに当たり，まずどういう組織構成にするか．また，工程の推移に応じてどう組織を変えるか，その正しさが成果を上げるためには重要なポイントとなる．

その場合，組織の最小単位は役割であり，各工程での成果が組織の課題となる．すなわち，各工程における役割を明確にし，成果物を明確に定義することである．しかる後，組織の最小単位を定義することになる．その際おのおの役割を全うするための知識・技能も明確に定義する必要がある．ただし，最低限必要な物のみで良い．システム開発においては，開発工程の切り替えは機敏に行う必要があり，役割を全うした組織は，素早く廃止し，新たな役割（次工程）を担う組織を構成する必要がある．この工程の推移に伴う組織の変化は，知識・技能の入れ替えであり，組織構造の変更と捉えるべきではない．組織の構造は，成果に対する貢献の違いにより設計されるべきであり，次の4種類から組織される（図3）．

① 最終成果物の生産を直接担当する部隊



当図の生産を直接担当する部門において，成果物はその時々により異なる．また成果物の品質保障は自らが行う役割である．

図3 貢献別の組織構造

Fig. 3 Classified contribution organization structure

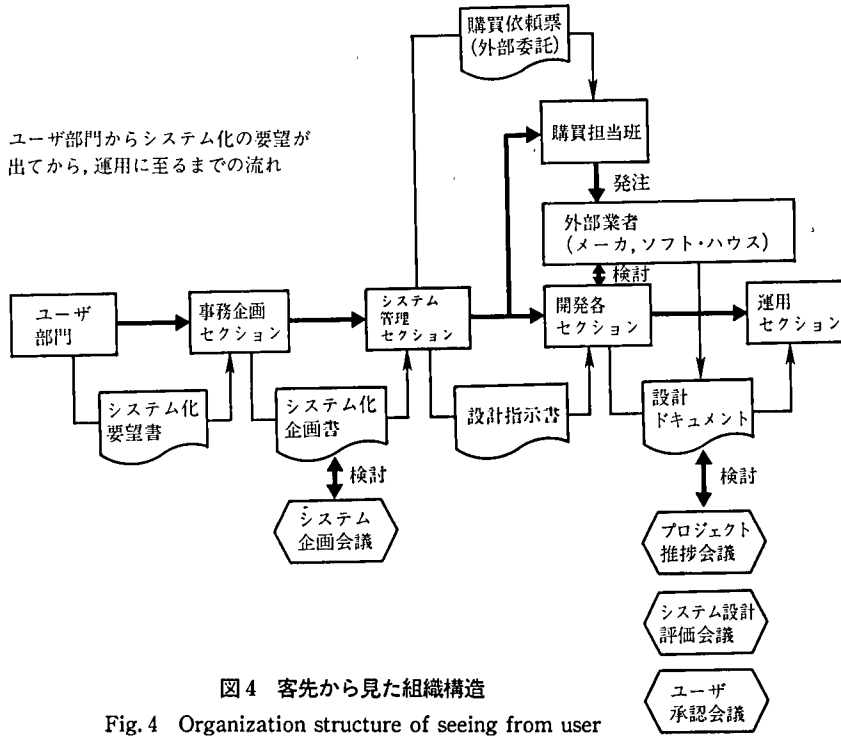


図4　客先から見た組織構造
Fig. 4　Organization structure of seeing from user

② 最終成果物の生産を支援する部隊
③ 最終成果物の品質を検証する，または正しい手続を踏んで生産活動が実施されたかを管理する部隊
④ 職場環境の整備，繁雑な事務処理等，生産活動とは直接的につながりのない活動を行う部隊

次に組織の規模についてであるが，仕事量または人間の数で組織の規模を考えるべきではなく，組織の構成・規模については次の観点で考えるべきであろう．

成果物に対する貢献の違う組織を一緒にすべきでない．逆に同一の貢献を果たす活動は同一のマネジメント下に置くべきである．ただし，マネジメントの限界を配慮すべきであり，マネジメントの限界は仕事および人的なボリュームの大小ではなく，関係を持つ他の組織とのチャネルの数によりおとずれる．つまり折衝すべき相手の数および種類により，マネジメント限界がおとずれる．マネジメントの限界を越えた場合は，同一の貢献を果たす組織であっても分けざるを得ない．

4. 仕　　事

4.1 仕事の定義

仕事とは一般的には，個々の作業または複数の作業を指すが，作業は仕事の一要素に過ぎず，課題達成のためのすべての行為を言う．したがって，課題達成のために必要となる知識・技能の修得も仕事と捉えるべきである．たとえば，一括発注先の組織において，知識・技能修得のための勉強会が持たれたとしたら，それは歓迎すべきことである．

いずれにしても，課題達成に向けてのプロセスとして正しければ良いのである．また，仕事をするには明確な課題つまり，各工程における最終成果物（いつまでに，何を）が，明示されていなくてはならない．課題・役割・責任が明確でないと人間は仕事をする意欲を失う．

仕事が課題達成へのプロセスを指すのであれば，それは生産的でかつ計画的でなくてはならず，この計画との差異が定量的に把握できる仕掛けがプロセスの中に組み込まれている必要がある．ただし，差異把握のためのプロセスが非生産的な行為に結び付かないように配慮すべきである．

マネジメントにおいては計画とのさまざまな差異により，プロセスおよび計画の変更を余儀なくされても，それを可能にする手段を前もって準備しておく必要がある．

4.2 仕事の生産性

システム開発において最終成果物はシステムであり，材料は顧客の具体的ニーズである．また生産に必要な道具は知識と技能である．顧客のニーズを理論的に組み立て，知識と技能を道具として使用し，システムを生産することが仕事であり，生産的な仕事を追求するには，材料と道具，それと生産活動に従事する人間について検討する必要がある．

1） 材料としての顧客の具体的ニーズ……顧客のニーズを基にシステムの生産がなされる．この大本となる材料が明確に定義されていないとか，生産過程で変更されるようなことがあっては成果につながらない，無駄な生産が営まれることになる．

2） 道具としての知識・技能……知識・技能は，その使い道いかんで最も高価なものにつく．つまり，効率的な生産を行うためには生産に必要な最小限の知識・技能を適用すべきであり，潤沢な知識・技能の動員はかえって組織を沈滞させることになる．またシステム開発においては，各開発工程により必要となる知識・技能は変わる．

3） 人間……人間は同じ動作の繰り返しに対しては，その能力は徐々に減退する．また人間は仕事を通じてその能力を成長させる．つまり，成長に応じ仕事の内容を高度化させることが最も望ましい．生産に必要な最小限の知識・技能は，人間の成長により確保することが最良であり，効率的な生産に結び付く．

5. 人　　間

5.1 資源としての人間

人間を物・金と同列の資源として捉えるべきではない．物・金は，生産活動の経費であるが人間は与えられた権限の範囲で物と金を使い成果をあげる有限，かつ限りない可能性を極めた資源なのである．したがって，資源としての人間を経費・労務の面で管理するのではなく活動を生産的足らしめるべく指導すべきであり，彼等の仕事の成果としての業績に目を向けるべきなのである．

また，彼等は責任を与えられることを望んでおり，仕事の結果に対し責任を負わせるべきである．責任ある仕事を行うには，その職務を遂行するための権限を必要とする．つまり，責任と権限は表裏一体の関係にあり，責任を負わせ権限の委譲がないと

いうことはあり得ないのである.

責任ある仕事をしようとする者は,自主的であり上司に対しあらゆる要求をする.時にはエゴイズムをむきだしにする場合すらある.それで良いのであり,責任集団の上に立つ者はそうでない者よりも,つらく厳しいのである.

職務の範囲を細かく定義することは,限りない可能性を持つ資源としての人間から成長の機会を奪い,職務と能力のミスマッチを引き起こし,当人の強みを発揮させる場をなくしてしまう.職場は挑戦に満ち,その職にあるかぎり成長と発展があるような,やりがいのある所でなくてはならない.組織は人のやる気で支えられ,やりがいが,やる気の裏付けである.

従前で述べたことは自社社員のみならず,システム開発過程での開発に携わった人々,全員に対してマネジメントが考えることである.人を利用して成果を手にしてはならない.人を活用し共に成果を手にするように心がけるべきなのである.

利用とは人の犠牲により成果を得ることであり,活用とは人の利益と共に成果を得ることである.

5.2 目標と自己管理

自己管理とは自分の役割を認識することから出発する.役割とは組織の目標に対して自分は何をせねばならぬか,他に何を期待することができるか.他から何を期待されているかを明らかにすることから出発し,役割としての課題に対し,自分の仕事ぶりや成果を生かすことなのである.また,個人の目標,組織の目標,さらにその上位の組織の目標は,同じ方向性を持っていなければならない.

システム開発に携わる組織体においても同様であり,客先・メーカ・ソフトウェアハウスの目標は同一の方向に明確に定義されている必要がある.さもなくば現場は烏合の衆と化し混乱する.

したがってマネジメントは,自分の権限が及ばない組織・人々(客先・メーカ・ソフトウェアハウス)であっても,その目標に問題があると認識した場合は,それを正す必要があり,目標管理の責任を持つ.

もう一つ,目標は高く挑戦的でなくてはならない.そこに技術革新と人間の成長がある.このリスキーな目標に対し成果を正しく評価し,成功へ導くことがマネジメントの真髄と考える.

6. マネジメント総論

6.1 古典的管理の否定

知的労働者組織においては,労務・経費の管理と言った古典的管理手法では現実の知識と理論による生産活動を管理することはできない.それは仕事の成果の管理であり,成果を評価することによる自己管理である.

6.2 プレイング・マネジメントとは

人間の強みを発揮させ,弱みを組織で補おうとすれば,広く挑戦に富んだ職務を若き部下に与えるべきであり,組織の持つ役割の一端をマネジメントの余力で行うべきではない.

マネジャがプレイする場は,自らの組織が受け持つ役割の範囲外とすべきである.

つまり，翌日に向けてのプレイを基本とすべきであり，若くて未熟な技能に対し寛大でありたい．

6.3 状況認識の重要性

高度に多元化された組織体をもって目標を達成せんとする今日，マネジメントにおいても自らの組織のみならず，他組織の目標達成のための責任を要求されつつある．この目標管理，すなわち正確な状況認識なくしてマネジメントはありえない．

正確な状況認識の基に的確な策が打たれる．策を行使するに当たり，コミュニケーション*が重要であることは言うまでもない．

6.4 革新の追求

世の中のニーズは常に少しずつ革新的であり，この革新への挑戦を忘れた組織は，いつしか大きなリスクを負って革新を断行するか，社会のニーズに応えられなくなり淘汰されることになる．活力ある組織は革新と挑戦に満ちた職場から生まれる．

また，マネジメントはその能力と責任において，あまり重要でない非生産性的なものは計画的に廃止し，革新への挑戦に振り向けねばならない．

7. おわりに

時代は今も新たな局面へと変化しており，過去の業績で得た知識・技術はすでに「陳腐化」しつつあり，また「陳腐化」させることが技術革新であり，生産的な組織・マネジメントである．つまり将来に向けて新たな組織のあり方，新たなマネジメントを追求，挑戦することこそ大事であり，顧客のニーズに応え顧客に満足を与えることにつながる．これこそ組織の目的であり責任であると信ずる．

本稿で述べたことは，筆者にとってのマネジメントの基本設計とも言うべきものであり，実施にあっての詳細については，今後整理していくつもりである．

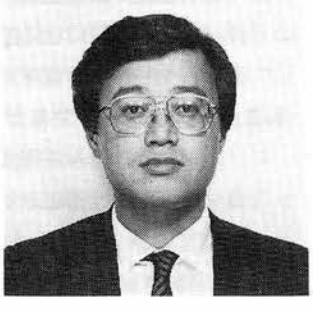
最後に私の日常の活動に，P・F・ドラッガー著「マネジメント」[1]の妙録である上田惇生訳出の「マネジメント」が非常に参考になっていることを付け加えておく．

また本稿の執筆に当たり，知識システム部 荒瀬部長，生産技術部 深堀部長のご指導に心から謝意を表したい．

＊ コミュニケーション：意思を伝達することではなく，聞き手の理解を求めることである．

参考文献 ［1］ P・F・ドラッガー著，上田惇生訳，「マネジメント」，ダイヤモンド社，1975.
［2］ D. マクレガー著，「企業の人間的側面」，
［3］ D. マーサー著，青木榮一訳，「IBM のマネジメント」，ダイヤモンド社，1988.

執筆者紹介 中 村 祥次郎 (Shojiro Nakamura)
昭和 29 年生．48 年滋賀県立彦根工業高等学校機械科卒業．同年日本ユニシス(株)入社．以後銀行の勘定系システムおよび証券系システムの開発に従事．現在金融システム本部金融システム二部に所属．



ISSN 0914-9996

技　報
UNISYS TECHNOLOGY REVIEW
Vol. 9　No. 3　(No. 23)

発　行　日　平成 元年 11 月 30 日
編　集　人　柳 生 孝 昭
発　行　人　富 田 和 夫
発　行　所　日本ユニシス株式会社
東京都港区赤坂 2-17-51 〒 107
TEL(03)585-4111 (大代表)
印　刷　所　三美印刷株式会社

UNISYS

100

100の、戦略。「100」の、力。

あらゆる業種の、あらゆる企業に、
戦略情報システムへの道を開きます。

戦略情報システムの核となる日本ユニシスのシリーズ2200に、小型汎用機2200/100シリーズが新登場。これにより、超大型から小型までシリーズ2200のフルラインアップが完成しました。シリーズ2200/1100のハードおよびソフト資産を継承しながら、手軽にお使いいただけるオフィス汎用コンピュータとして、さまざまな問題を速やかに解決します。

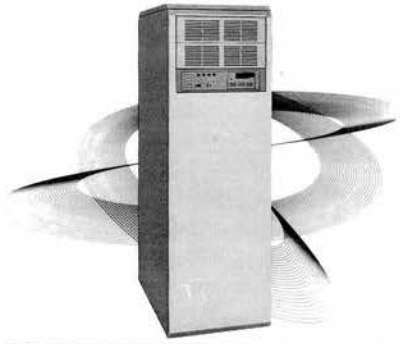
●システムの早期稼働を実現するために、豊富なソリューション・ソフトウェアを用意。定評あるSEサービス集団もいっそう充実。

●超大型機までシングルOSによって貫かれた2200シリーズ内での上位移行を保証。情報資産を活かしながらステップアップできます。

●分散処理ネットワーク・システム構築支援ソフトウェア「UNIDSS」によって、異機種間の分散処理ネットワーク・システムが構築できます。

●世界の2大4GL(MAPPER、LINC)が利用可能。エンド・ユーザー自らによるアプリケーション開発、データベース活用が実現できます。

●90年代に向けたSISのオフィス系システム構築をサポートするインフラストラクチャとして「EXOS」(エクソス)を提供します。



小型汎用コンピュータ
UNISYS 2200/100シリーズ

UNISYS 2200/100シリーズ新登場

日本ユニシス株式会社 本社 東京都港区赤坂2-17-51 〒107 電話03-585-4111(大代表)