TECHNOLOGY REVIEW

UNISYS 技報 通巻 41

1994年5月発刊 Vol. 14 No. 1

小特集：分散システムの一事例

日本ユニシス

ISSN 0914-9996

日本ユニシス（株）のオンライン事務処理として開発された営業情報システムもすでに十数年経過し，商内の拡大や変更と相俟って，従来汎用ホストと端末で構成されていたシステムからOSをUNIX* とするオープンプロダクトを採用した分散処理型のシステムに再構築した．平野吉延は**新営業システムの概要——典型的な分散処理システム**の中で，基幹事務処理システムとしての新営業システムの概要を報告している．

*UNIXオペレーティングシステムはUNIX System Laboratories, Inc.が開発し，ライセンスしている．

メインフレーム上のシステムのダウンサイジングを成功させるには，メインフレーム/サーバ/ワークステーションへの機能の分散とシステム的統合を実現する必要がある．そのためにはプラットフォームの違いやネットワーク等を意識したシステム開発を行わなければならず，開発者にとって大きな負担となる．小関洋・芝尾文彰の**クライアント/サーバ・コンピューティングを指向した新営業基盤システム**は，大規模分散システムの構築のために整備したデータ交換・印書などの基盤システムに対する技術的課題とその解決策を，データ交換・印書・統合メニュ・ネットワーク環境設定支援の紹介を通して述べている．

営業情報システムはメインフレームと端末による集中処理形態で運用されていたが，全国の各営業部門に26台のUNIXサーバと123台のワークステーションを配置する分散処理形態を採用する新システムに変更した．各部門では一般の営業部門のスタッフが運用管理者に選任され，基幹業務システムの運用の一部を担うことになった．内池貴明の**UNIXシステムの運用管理**は，基幹業務システムをUNIX上に構築する場合に運用面でどのような配慮が必要であったか，その実例を紹介している．

複数のUNIXサーバとワークステーションを営業部門ごとに設置する新営業システムでは，従来メインフレームコンピュータで集中管理していたデータやプログラムファイルを各UNIXサーバやワークステーションに展開することになる．添田和宏・宮越貞二は**3階層分散システムにおけるデータ配置とその維持**の中で，ファイルの分散化に伴いファイルの配置とデータの整合性を維持するために設計上どのような配慮を行ったかについて報告している．

事務処理システムのデータベースをUNIX上に構築する場合，ハードウェア，オペレーティングシステム，データベースソフトウェアが様々なメーカから提供されるため，それらの統合的なパフォーマンスを考えることは困難になっている．鈴木温雪の**データベースのチューニングの実際**は，メインフレーム上にあるデータベースを分散し，複数のUNIXサーバに多いもので数十万件のデータを格納する新営業システムの実例を通してそのパフォーマンス改善の実際を報告している．

原子力プラントのマンマシンインタフェースは操作スイッチや指示計の数で1000程度もある．このような運転環境は自動化の導入，情報のCRTへの集約表示等により複雑・多様化しつつある．電力中央研究所・ヒューマンファクタ研究センタでは，プラントにトラブルが発生した時のヒューマンエラーの発生過程を解明し，その防止策の立案に資するため，運転チームの意思形成過程を個人面，チーム面から検討できる運転チーム行動モデルの開発を進めている．吉村誠一・高野研一・佐相邦英は**運転チーム行動モデルの開発**の中で，モデル開発の全体計画ならびにモデルの概要について述べている．

電力中央研究所ヒューマンファクタ研究センタの委託により「発電プラント運転員チーム行動シミュレーションシステム」を開発した．チーム行動は，文脈に基づく認知・判断と会話によってモデル化されると考え，この点で有効な記述方法を持つOMTを開発法として採用した．羽田昭裕の**OMTを使った「発電プラント運転員チーム行動シミュレーションシステム」の開発**は，前論文の研究に基づくシミュレーションシステムを開発者の視点で記述している．

新営業システムの概要
——典型的な分散処理システム

An Overview of a New Marketing Information System
——for Typical Distributed Data Processing

平 野 吉 延

要 約 日本ユニシス（株）のオンライン事務処理システムとして開発された営業情報システムもすでに十数年経過し，商内の拡大や変更と相まってシステムを変更することとなった．これを機会に，従来汎用ホストと端末で構成されていたシステムからオペレーティングシステムをUNIX*とするオープンプロダクトを採用した分散処理型のシステムに再構築した．新しいシステムは，グループコンピューティングのインフラストラクチャとしても利用できるようLANを敷設し全国の支社支店および営業所をネットワークで繋いでいる．また新システムは，従来の汎用ホスト集中管理から分散運用形態に変更になるため，分散するコンピュータで稼働する業務システム同士を連動させるための仕組みや各部門の運用を支援するための機能も合わせて開発した．本稿は基幹事務処理システムとしての新営業システムの概要を報告する．

Abstract It has already been ten-odd years since the marketing information system developed by Nihon Unisys for on-line business data processing made a debut. However, extended volumes of business deals and changes in the way of doing business have forced the system to be rebuilt into a UNIX OS-based distributed system, for which open systems products are adopted, from the conventional network system linking a mainframe with terminals. So designed as to provide an infrastructure of group computing as well, the new system connects together all of the regional offices, branches and sales offices across the country through local area networking. Additionally, in order to further enhance the new system which entails a change in the mode of operation from centralized host-oriented data processing to distributed computing, the development efforts also included the creation of a mechanism to link applications run in the decentralized data processing environment as well as of the function to support operation at individual departments. This paper gives a thumbnail sketch of the new marketing information system positioned as one of the systems to process mission-critical business applications.

1. はじめに

日本ユニシス（株）の営業情報システムは，営業活動の結果である受注・契約から解約までの営業情報を管理するとともに，発注・納品等の物流管理や会計処理など，後続の基幹システムのフロントシステムとしても位置づけられる．近年，取扱う商品やサービスの拡大に伴いシステム再構築の要望が出されたのを機会に，メインフレームと端末で集中オンライン処理していたシステムをオープン化に即応して営業部門毎にサーバとワークステーションを配備した分散処理形態のシステムへと再構築した．

一方，営業の事務処理を合理化および簡素化し，本来の営業活動への注力を強化す

*UNIXオペレーティングシステムは，UNIX System Laboratories, Inc.が開発し，ライセンスしている．

ることがますます重要になってきている．そのためには，事務処理手続きそのものを見直して変更することと，それをシステムで支援することが必要となる．新システムの第一の目的は，この事務処理を支援することである．第二の目的は，営業部門の「情報化武装」である．これは営業部門の計数管理という面と，販売するために有益な情報を提供することである．

営業部門の計数管理は，従来のホストシステムでは利用者側は必要な情報を情報システム部門やスタッフ部門に依頼して入手することとなり，自主的に必要な管理資料を作成することが困難であった．そのため，新システムに於いては利用者が随時必要な情報を抽出可能とする環境を提供することも一つの目的である．また，販売のための情報という点では商品の技術情報や顧客に関する情報等をサーバより取り出し，加工できるようなインフラストラクチャを用意することである．

そして第三の目的は，最新技術を駆使してシステムを構築することにより技術の集積と蓄積をはかること，および当社商品の有用性を実証することである．

本稿では，従来汎用ホストと端末で構成されていたシステムを以上の目的にそって分散システムとして再構築した新営業システムの概要について記述する．

なお，本稿のほかに新営業システムに関する次の論文も併せて参照されたい．

- 「クライアント/サーバコンピューティングを指向した新営業基盤システム」（小関洋・芝尾文彰 共著）
- 「UNIX の運用管理」（内池貴明 著）
- 「分散システムにおけるデータの整合性とその維持」（添田和宏・宮越貞二 共著）
- 「データベースチューニングの実際」（鈴木温雪 著）

2. 分散処理検討

システムを分散化するに当たり，従来のホスト集中処理の営業システムでの問題点および利点を整理してみる．

問題点

① ホスト上で稼働する他システムの処理負荷に影響される．

② エンドユーザが管理資料を作成しようと自部門のデータを加工する場合，抽出に要する時間や加工のためのソフトウェアの制限など，種々の制限を受ける．

③ 利用時間帯が利用部門の意のままに必ずしもならない．

利　点

① システムの運用が集中管理できるのできめ細かな統一した管理が可能である．

② エンドユーザが利用するのに専門的知識を必要としない．

③ システム構築の際に開発技術が比較的成熟しているので，安定している．

分散化を検討するに当たり，上記の問題点を解消し，なおかつ分散化で失われてしまいがちな利点をできるだけ維持できるようにシステム化を検討した．利点①については，全システムをコントロールする運用管理サーバを設置して，中央からのコントロールにより統一管理を目指した．利点②については豊富なメニューおよび HELP 機能により操作の容易化をねらった．

新システムでは一般化された最新技術を最大限に採用し，従来のホストシステムでは実現できなかった情報処理環境を提供することとした．したがってサーバおよびワークステーションのOSの選定に当たっては

- グラフィカルユーザインタフェース
- マルチウインドゥ
- マウス等のポインティングデバイス
- ETHERNET*
- クライアント/サーバ

などが利用できるものでなくてはならない．

候補としては

- UNIX
- OS/2**
- WINDOWS**

があげられたが，当社の主力製品であるUNIXをサーバ，ワークステーションとも採用することとした．このことにより，プログラムはサーバ，ワークステーション間で移植性（互換性）が得られることとなった．

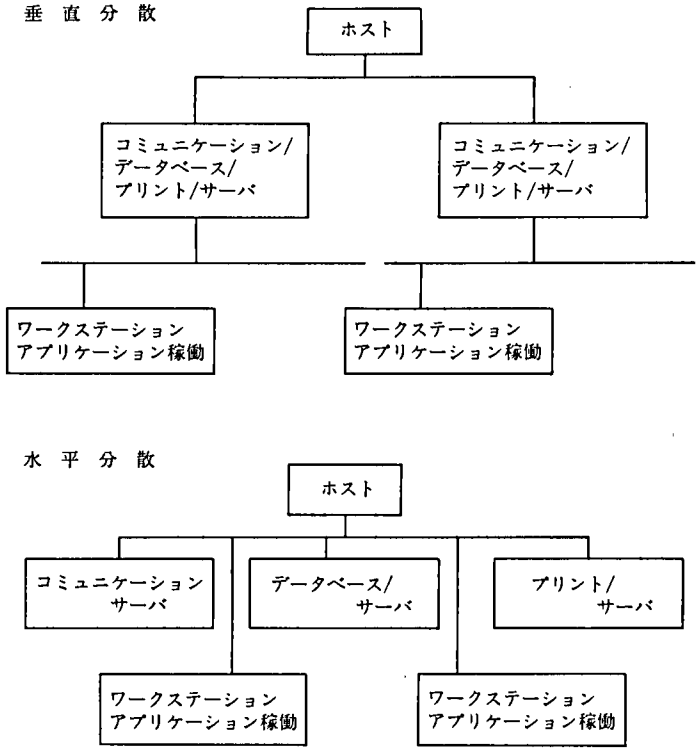


図 1　垂直分散および水平分散

* ETHERNET（イーサネット）は米国Xerox社の登録商標である．
** OS/2，WINDOWSは米国Microsoft社の商標である．

分散化の形態として，機能および負荷を垂直に分散するシステムと水平に分散するシステムとが考えられる(図1)が，次のような点から，垂直分散の方がより新システムに適していると判断した．

- ネットワークが全国規模となり，地方の場合，サーバを各機能ごとに配備することは非効率となる．
- データベースサーバは，担当する範囲が広がるほど，アクセス効率が悪化する．したがって，1サーバ当たりの収容データベースをできるだけ少なくする．

3. システム概要

3.1 ハードウェア構成

新システムのハードウェア構成は図2に示す通りである．

- メインフレームホスト－UNISYS1100/92

  従来の営業システムおよび後続の物流・金流システムが稼働しているホストである．新システムにおいては，サーバから入力されたデータはファイル転送にて，ホストに送出され，データベースとして蓄積され，後工程システムへのインプット，および全社管理情報の検索・加工へと利用される．

- 部門サーバ－U6000/60

  本社17か所，支社・支店9か所の計26か所に設置されたサーバは，データベースサーバ，コミュニケーションサーバ，プリントサーバの役割を担う．とくにデータベースサーバとして，各部門の営業商内データをORACLE*上に展開していて，一般利用者は自部門データのみのアクセスが許されている．

- 各サーバはホストと通常の回線経由で接続される．

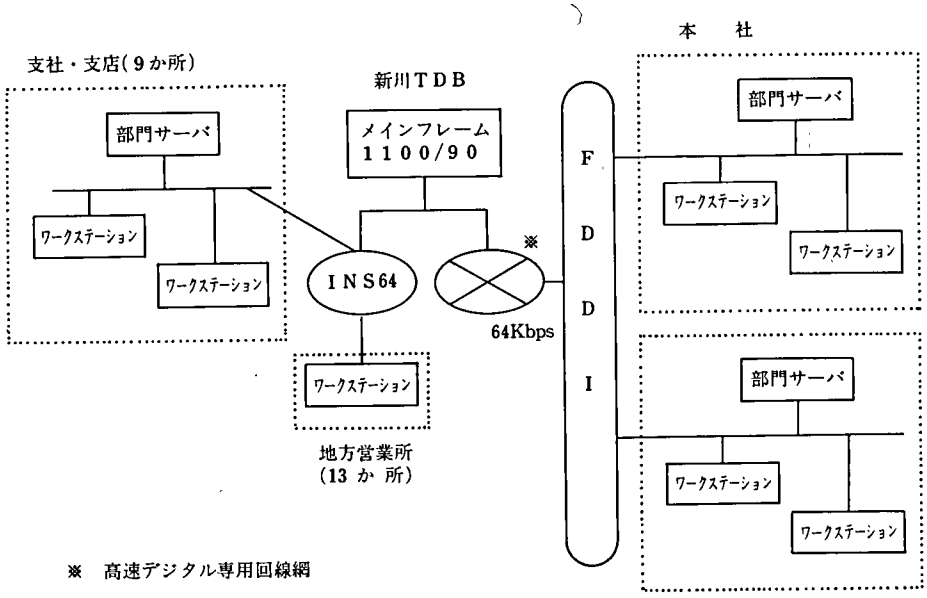


図 2　ハードウェア構成図

* ORACLE は米国 ORACLE 社の登録商標である．

新システムにおける新機能の一つとして契約書の印書出力があり，そのための印書装置を各サーバに2種類接続している．また，データベースは一日単位で障害時のためにバックアップテープに吸い上げを行うので，カセットテープ装置を接続している．

・ワークステーション－U6000/DT2

各サーバに数台LAN経由で接続される．システム全体で約130台のワークステーションが設置されている．地方営業所13か所にはワークステーションのみ1台設置され，どこかのサーバに通信回線経由で接続されている．

3.2 ソフトウェア構成

新システムのソフトウェア構成を図3に示す．

・オペレーティングシステム

サーバ，ワークステーションともUNIX SVR4を搭載している．ホストはOS/2200である．

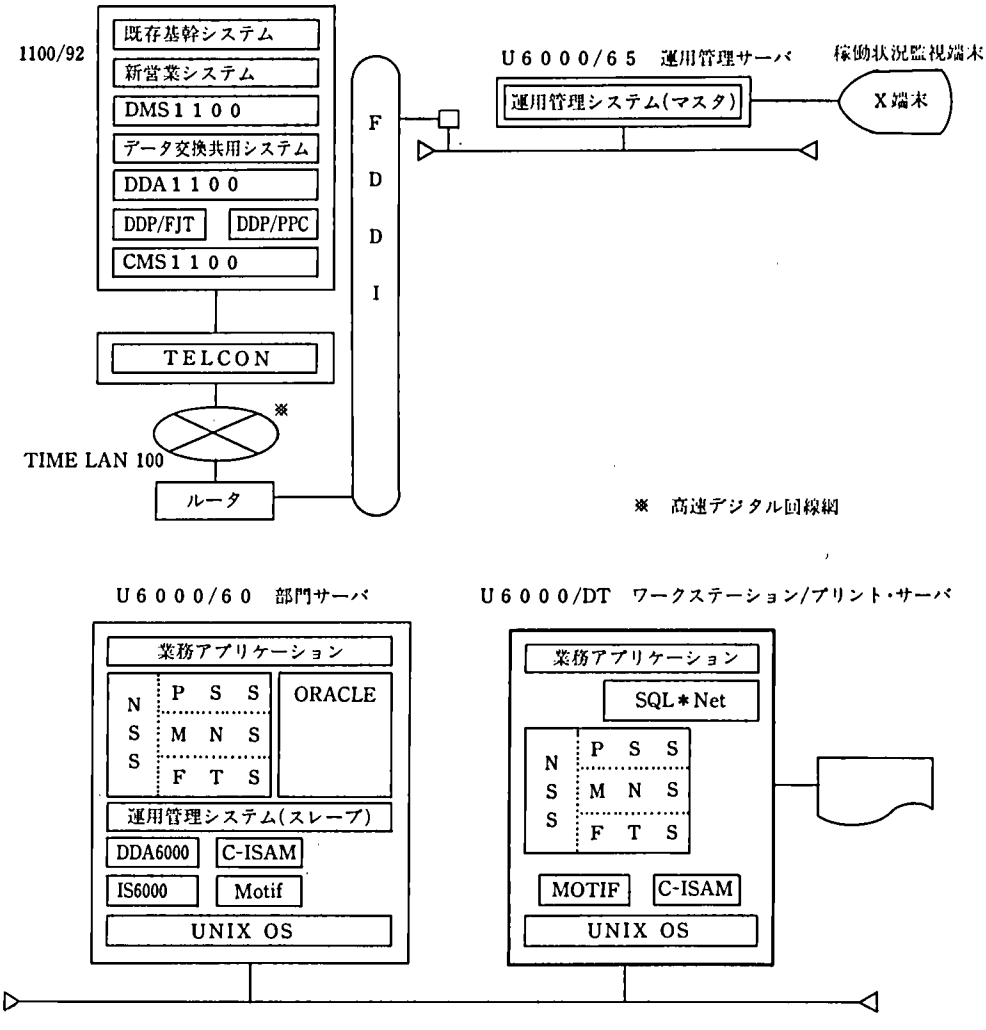


図3 ソフトウェア構成図

・データベースソフトウェア

ホストのDMS1100が全社データベースを保持する.

・サーバ—ORACLE, C-ISAM*

部門データベースをORACLEデータベースで保持する.また,共通マスタコード等の検索用データベースはC-ISAMで保持する.

・ファイル転送関連ソフトウェア

DDA1100（ホスト), DDA6000（サーバ）の基本ソフトウェアを業務アプリケーションが容易に利用できるようにデータ交換共用ソフトウェア（FTS）と名づけたミドルソフトウェアを開発した.

・印書関連ソフトウェア

当社のACCFを具現化したプロダクトの一つである分散プリントを使い,さらに利用者の印書装置操作を容易にする目的でPSSと名づけたミドルソフトウェアを開発した.

・運用管理関連ソフトウェア

各部門サーバとは別に全システムの運用管理を司る運用管理サーバを導入し,統合運用管理システム（OMEGA）を稼働させている.この監視サーバをマスタと呼び,配下の部門サーバを集中的に監視するとともに配下のワークステーションの稼働状況も監視する.また,プログラムおよびファイルの配布のためのソフトウェアNSSを開発した.

・業務アプリケーション

業務アプリケーションプログラムのほとんどはC言語で組まれ,ワークステーション上でクライアントとして稼働する.また,GUIとしてMOTIF**を利用している.

4. 本システムの特徴

本システムの特徴を列記すると下記の如くなる.

・ホスト/サーバ/クライアントの3階層システム

・UNIX/UNIXのクライアント/サーバシステム

・無人運転に近い各部門の運用管理

1) 3階層の分散処理システム

3階層に分散化し,かつ,すそ野に広がるにつれ機器数の増加するシステムにおいては以下の事項が課題となる.

・各部門データベースと全社データベースの整合性の維持

・プログラムおよび共通コードファイルの同一性の保証

・各部門での一定枠の自由な運用管理（運用時間, セキュリティ等）の確保

これらの課題の解決については,本誌別稿の「分散システムにおけるデータの整合性とその維持」で記述される.

2) 同一OSのクライアント/サーバシステム

---

* C-ISAMはINFORMIX Software社の登録商標である.

** MOTIFはOpen Software Foundationの登録商標である.

新システムのサーバとワークステーションのOSはUNIXである．一つのプログラムはサーバ，ワークステーションどちらでも稼働可能であるという移植性がある．このことから，システム設計，プログラム設計を行うに際して，以下の利点がある．

・アプリケーション・プログラムは特別にクライアント/サーバを意識せずに設計ができる．クライアント/サーバのためのプロセス間通信を司るミドルソフトウェアがこの違いを吸収する．
・サーバの処理能力が不足する場合，サーバの増強をせずに余裕のあるワークステーションに一部のサーバ機能を移植する，またはワークステーションを追加することにより対処可能である．
・地方営業所の場合，サーバとワークステーション間がINS回線であり，本社と同様の役割分担にした場合，サーバ，ワークステーション間のデータの授受に時間を要し，レスポンスタイムが限度を越える．したがってサーバ機能の一部または全部をワークステーションに移植することで，小規模営業所にサーバ用としてのハードウェアを設置しないですむ．
・X端末でも稼働可能であり，とくに開発時のように処理負荷が問題とならない場合に，安価なX端末でも開発が可能である．

3)　無人運転に近い各部門の運用管理

新システムの運用管理で最も大きな課題となるのは全国の利用部門に設置された26台のサーバ，LANおよびワークステーションの運用管理である．特にサーバは各部門のデータベースを保持するため，小規模ながらバッチ処理が必要であり，汎用機と同様の運用管理が必要となる．しかも利用部門は営業職であり，運用管理のための専門知識を強要することは不可能である．そのため，情報システム部門に運用管理サーバを設置し，ほぼ無人に近い運転および運用管理を実現した．そのための実現項目は，

・部門サーバの立上げ，終了処理の自動化
・サーバおよびワークステーションで必要となるソフトウェア（プログラムおよび各種ファイル）配布・反映の自動化
・サーバ，ワークステーションの障害情報の運用管理サーバ，運用監視ワークステーションへの通知，およびロギング

これらの詳細については本誌別稿の「UNIXの運用管理」で記述される．

5. おわりに

今回のシステムは新しい技術，ソフトウェアを用いた分散オープンシステムという点で未経験の技術上の問題点を克服しながら，システム構築を行ったが，以下の点が課題として残されている．

1)　アプリケーション・プログラムはすべてC言語で記述したが，必ずしも開発生産性が良くなかった．開発開始時点ではU6000上で稼働するTIPPLERが未提供であったが，今後TIPPLERの採用の検討が必要である．
2)　メインフレームとサーバ間はファイル転送を用いたが，一部リアルタイム性が

要求される局面があり，今後 OLTP の検討が必要である．

開発目的に照らして今回のシステムを評価すると

① 事務処理の効率改善

事務処理手続きの大幅な変更・改善が一部なされないままシステム化された点と，画面設計において旧来の入力画面にとらわれた点とがあり，今後改良・改善に努めなくてはならない．

② 営業部門の情報化武装

今回のシステム化により環境基盤が整った．今後各種ソフトウェアツールを提供していくことにより，利用部門が容易に自ら必要な情報の入出力を可能としたい．

③ 技術の集積と蓄積

開発当初，要員の大部分が UNIX，オープンに関しての知識がなく，したがって組織としても対応力が欠けていた．開発が一段落した現在，オープン化にまつわるソフトウェア技術に関しては単に知識のみでなく，経験の蓄積がなされたことにより，組織としての対応力が強化された．

今回のシステム構築に当たり，初期の納期と品質を守るため日夜努力をされた情報システム部員と利用部門の指導に当たられたマーケティング部員に謝意を表したい．

執筆者紹介 平 野 吉 延 (Yosinobu Hirano)

昭和 18 年生．41 年東京大学農学部農業工学科卒業．43 年日本ユニシス(株)入社．証券・保険会社のシステムを担当．以降 CS 部門，プロダクト主管部門，金融システム部門を経て，現在 情報システム部長．

クライアント/サーバ・コンピューティングを指向した新営業基盤システム

A Client/Server Computing-oriented New Infrastructure System for Marketing Information Processing

小　関　　洋, 芝　尾　文　彰

要　約　メインフレーム上のシステムのダウンサイジングを成功させるには, メインフレーム/サーバ/ワークステーションへの機能の分散とシステム的統合, すなわちクライアント/サーバ・コンピューティングを実現する必要がある. しかし, そのためにはプラットフォームの違いやネットワークや通信プロトコルを意識したアプリケーションシステムの開発を行わなければならない. これは, 開発者にとって大きな負担となる. そこで新営業システムでは, データ交換・印書などのアプリケーション構築の基盤となる機能を基盤システムとして整備し, 前述のような難しさを意識しなくてもよい開発・実行環境をアプリケーションシステムに提供した.

本稿では, まず新営業システムの特徴を示し, 基盤システムへの技術的課題を明らかにする. 次に, 基盤システムを構成するデータ交換, 印書, 統合メニュー, ネットワーク支援の各システムの紹介をとおして, それぞれに課せられた技術的課題とその解決策をUNIX* システムおよび, ミドルソフトウェアの利用技術という観点から述べる.

Abstract　Successful system downsizing on the mainframe requires functional distribution to the mainframe/server/workstation and systems integration, that is, the implementing of client/server computing. To meet those requirements, however, applications developers need to do their jobs, conscious of differences of platforms and network/communications protocols. This has been a big headache to those people. Then, a new marketing information system was so created as to support basic functionalities required for applications development such as data exchange, printing and the like, providing a new development and operational environment that helps systems creators work without paying special attention to such difficulties as listed above.

Besides referring to the specific features of the new marketing information system, this report clarifies technical requirements for the infrastructure system. Also, through the mentioning of the system's elements that are file transfer, print support, menu selection for integration and network support, this paper discusses what challenges the authors still have to contend with for enhancement of each component and conceivable individual solutions, focusing on the adoption of a UNIX-based system and on the use of middle software.

1. はじめに

サーバ/ワークステーションの高性能化・低価格化, OSI や TCP/IP などの通信技術の発展と普及を背景として, メインフレーム集中型システムの分散システム化, いわゆるダウンサイジングが検討され, 実行されつつある.

* UNIX オペレーティングシステムは UNIX System Laboratories, Inc. が開発し, ライセンスしている.

ダウンサイジングは，負荷分散によるシステム全体のスループットの向上とコスト低減を目的としたものである．しかし，分散したサーバに従来のメインフレームと同じ役割を負わせるだけではダウンサイジングのメリットを享受できない．トランザクション量の増加等により，サーバがいずれは現在のメインフレームと同じ運命を辿るからである．

分散システムのメリットを十分に出すためには，メインフレーム/サーバ/ワークステーションでのクライアント/サーバ・コンピューティングを実現する必要がある．しかし，そのためには，プラットフォームの違いやネットワークや通信プロトコルについての知識が必要であり，これらは開発者にとって大きな負担となる．新営業システムでは，プラットフォーム間でのデータ交換や，ネットワーク内の任意の印書装置への印書などのアプリケーション構築の基盤となる機能を基盤システムとして位置付け整備した．基盤システムを使うことにより，前述のような分散システム特有の難しさを考慮することなく，クライアント/サーバ型のアプリケーションシステムを容易に構築することができる．

本稿では，大規模分散システムである新営業システムの基盤システムに対する技術的課題とその解決策を，データ交換 (4 章)，印書 (5 章)，統合メニュー (6 章)，ネットワーク環境設定支援 (7 章) の紹介を通して述べる．さらに，本番稼働 6 か月を経た現状をふまえて，基盤システムの評価と今後の課題についてふれる．

2. 新営業システムの特徴と基盤システムへの課題と目標

新営業システムは，以下の特徴を持ったシステムである．

1) メインフレーム，UNIX サーバ，UNIX ワークステーションからなる 3 階層の分散処理システムである．
2) メインフレームは，分散システム内に 1 台だけ存在する．実用上十分なセキュリティ管理機能のもとで，専任のオペレータが既存基幹システムを運用する．
3) UNIX サーバを，部門ごとに 26 台設置する．運用は，専任ではない部門の運用管理者が行う．アプリケーションのデータベースを管理する．
4) UNIX ワークステーションを，部門ごとに数台設置する．運用は利用者にまかされる．アプリケーションの実行を行う．
5) メインフレームと UNIX サーバ間は，64 Kbps の高速デジタル専用回線で接続する．地域によっては，INS 64 の回線交換で接続する場合もある．
6) UNIX サーバと UNIX ワークステーション間は，10 Mbps のイーサネット* LAN で接続する．ただし，本社のワークステーションから地方のサーバをアクセスする場合は，64 Kbps の WAN を経由する．
7) アプリケーションは，メインフレーム上のリアルタイムシステムの作り直しであるため，利用者・開発者ともにメインフレーム・端末型のシステムのイメージを強く持っている．

基盤システムに対する課題は，次のものである．

1) メインフレーム上の既存システムと UNIX サーバ上のシステムとのデータの

* イーサネット (ETHERNET) は米国 Xerox 社の登録商標である．

交換を行う．そのためのトランザクション処理を実現する必要があるが，メインフレーム上のリアルタイムシステムほどのパフォーマンスは実現できなくても良い．

2)　メインフレームが稼働していなくても，UNIX サーバが運用できること．

3)　自部門内の任意の印書装置へ印書でき，操作性は旧システムと同等以上．

4)　アプリケーションの開発・運用は，実行プラットフォームの違いを意識しなくてすむこと．

以上の課題は，2)を除いて一言で言うとメインフレームのシステムと同等の機能を実現してほしいということである．メインフレーム上のシステムのダウンサイジングでは，過去のイメージを強く引きずるのも止む終えないところであるが，基盤システムとしては，将来の EUC の発展や新営業アプリケーション以外のことも考慮しなければならない．今回開発するアプリケーションシステムは，垂直分散型のシステムであるが，基盤システムとしては水平分散型のシステムにも対応できる必要がある．そこで，次のように目標を設定した．

・各プラットフォームの特徴を生かした分散協調処理基盤を構築する．すなわち垂直方向・水平方向への負荷分散と，メインフレーム/UNIX サーバ/UNIX ワークステーションでのデータ連携を中心とした協調処理の基盤構築をめざす．

3. 新営業基盤システム概要

基盤システムは，互いに独立の四つのシステムから構成されている（図 1)．

① FTS（File Transfer System：データ交換共用システム）
メインフレームと UNIX サーバ間でのデータ交換を実現するシステム．

② PSS（Print Support System：帳票出力支援システム　）
分散された UNIX 環境での印書を支援するシステム．

③ MNS（Menu System：メニューシステム）
GUI の画面からマウスによる選択だけで，あらゆるアプリケーション・システムを起動できるようにするためのシステム．

④ NSS（Network Support System：ネットワーク支援システム）
分散環境下でのネットワーク環境の設定と，マシン間での通信の支援を行うシステム．

これらは垂直分散型，水平分散型のいずれのシステム構成もとりうる（図 2)．

メインフレームでは，アプリケーションシステムが必要とする全てのサービスは物理的に一つのコンピュータの中にあり，オペレーティングシステムを始めとする基本ソフトウェア群が全てのサービスを制御している．ところがクライアント/サーバ型のシステムでは，サービスの提供者がネットワーク内の複数のプラットフォームに分散して存在する．アプリケーションシステム（クライアント）は必要とするプラットフォーム（サーバ）を選択し，サーバに応じた手段でサービスの提供を受けなくてはならない．

新営業基盤システムでは以上のようなクライアント/サーバ型システム特有のアプリケーションシステム開発および実行時の負荷を軽減し，アプリケーションシステム

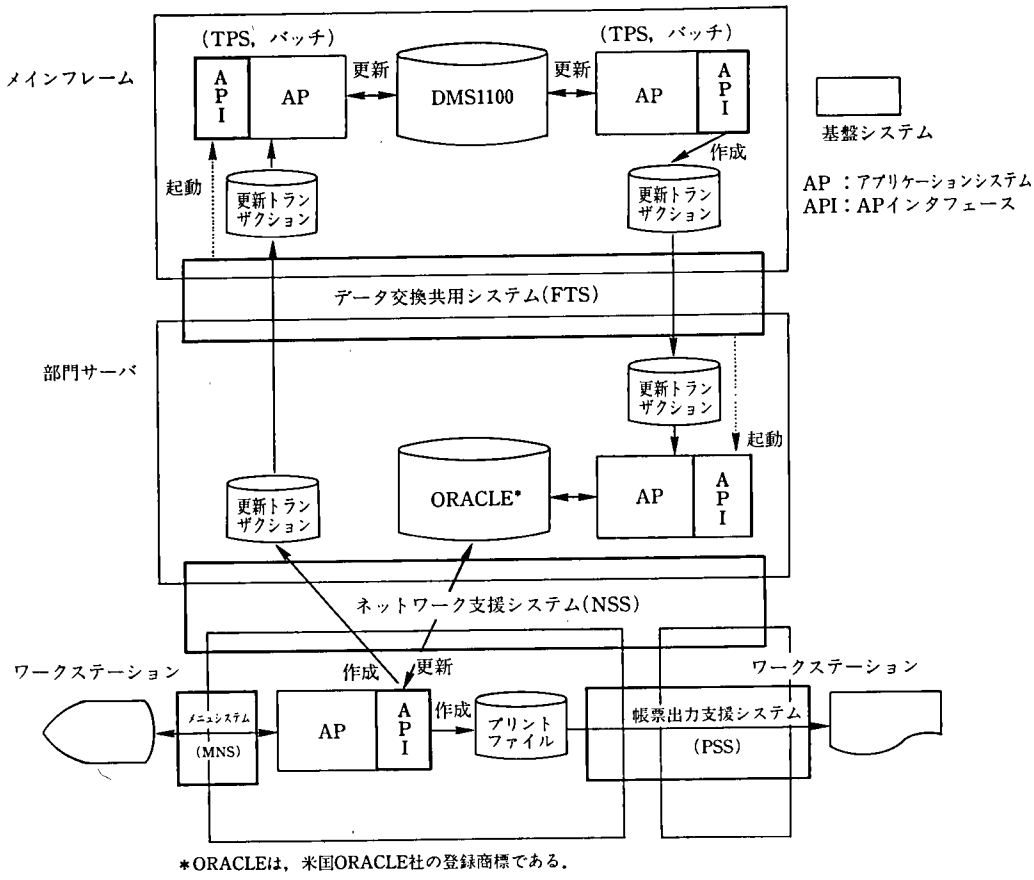


図 1　基盤システム概要図

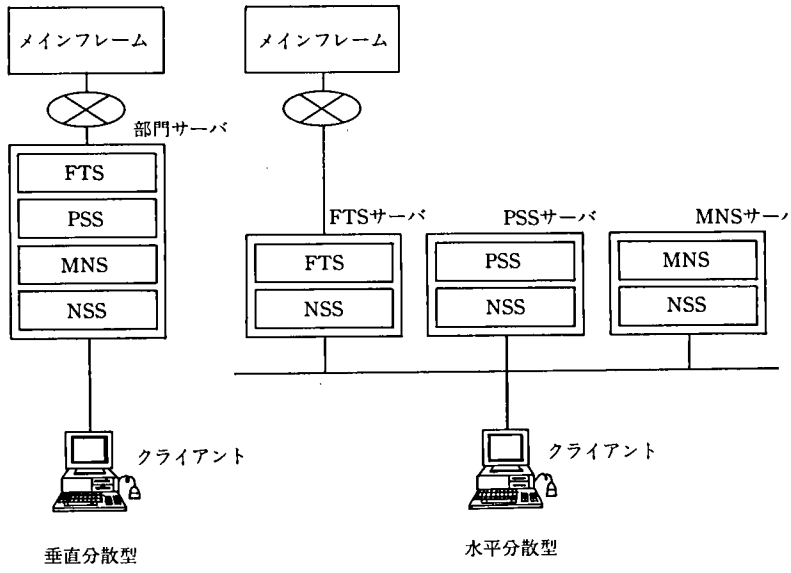


図 2　基盤システムの適用形態

がデータベースの更新のような独自の処理のみに専念できる環境の構築を試みた．ネットワークでつながった複数のプラットフォームが，あたかも論理的に一つのコンピュータのように扱えるような環境の提供を目指したのである．

4. メインフレームとサーバとのデータ交換（FTS）

4.1 要　　求

アプリケーション開発者側から見た要求は以下のとおりであった．

1) 1日1000回程度のデータ交換を行う．
2) 1回に交換するデータは，800～250 Kバイト，1000バイト程度のものが多い．
3) コードマスタを毎朝メインフレームからサーバへダウンロードする．
4) 日中はメインフレーム上のアプリケーションとサーバ上のアプリケーションが双方向のデータ交換を行いながら連携して動く．
5) 転送先が非稼働でも転送指示は受け付ける（夜間および日祭日等）．
6) アプリケーションのターンアラウンドタイムは，従来の営業システム（リアルタイムシステム）と同等かそれ以上である．

4.2 要求分析と目標設定

以上のような新営業システム独自の機能要求を，既存の通信ソフトウェア・プロダクトのみで実現することは不可能である．これには，通信ソフトウェアを組み合わせ，WAN/LAN等のネットワークに接続されたメインフレームとUNIXサーバがあたかも一つのコンピュータであるかのように，ユーザがそれらを意識することなく処理し，各部門サーバごとにメインフレームの稼働，非稼働に影響されることなく独立分散した運用を行い，またデータ転送アプリケーションの設計・開発・保守の容易性を主目的としたミドルソフトウェアの開発が必要である．

データ交換の方法として，プログラム間通信型とファイル転送型の二つが考えられる．プログラム間通信型は，比較的データ量が少なく，緊急性の厳しい，リアルタイムで連携処理を行うような用途に向き，交換頻度が1～数回/日で，データ量が多い場合には，ファイル転送で送るのが現実的である．要求の1)を満たすためにはプログラム間通信型を採用するのが現実的ではあるが，要求の2)を実現するにはファイル転送が有効であると考えられる．

新営業システムでは，比較的緊急性が要求され交換回数も多いが，1回のデータ量が大きいこと，あるいはアプリケーションの転送要求に対するACKの応答時間を短縮しなければならないことから，相手先と密接に連携するプログラム通信よりもファイル転送によるデータ交換システムとした．要求の1)についてはファイル転送を頻繁に実行することにより解決できる．また，メインフレームと部門サーバに同等の機能を持たせ，メインフレーム ⇄ UNIXサーバ間，UNIXサーバ ⇄ UNIXサーバ間，およびメインフレーム ⇄ メインフレーム間(新営業システムではメインフレームは1台である)でのデータ転送を標準処理パターン（図3）として提供するために，組み合わせる転送ソフトウェアにより3) と4) の要求は解決できる．しかし，転送先の稼働状況に関係なく転送要求を受け付け，アプリケーションのターンアラウンドタイムを短縮し，リアルタイム性の高いデータ交換を行うという要求の5) と6) は難しく，転送要

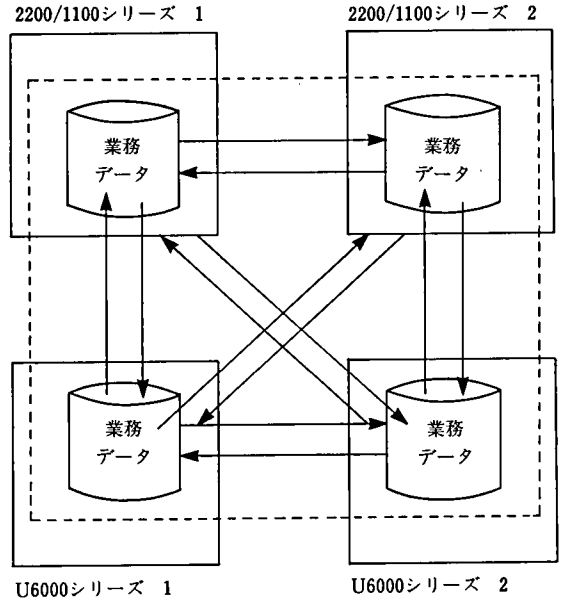


図 3 データ交換パターン

求の受け付けと実転送を同期をとったのでは不可能である．これは転送要求を一旦バッファリングし，その直後にファイル転送を行うことにより実現可能である．

これらの要求の実現は不可欠ではあるが，このような基盤システムを開発する上でもう一つ不可欠なのが拡張性・移植性の考慮である．今回採用する通信ソフトウェアも現在出されている要求だけでなく，今後システムが複雑化する傾向にある中で，機能の拡張または高水準のファイル転送への変更は避けられない．そのため，より高性能な転送ソフトウェアへ移行できるような設計でなければならない．同様にハードウェア・ネットワーク構成においても，環境によっての変化は否めず，移植性に富んだ協調性のあるシステムを目指さなければならない．

以上のことを考慮して，次のように目標を設定した．

1) DDA (Distributed Data Administration system : 統合データ交換システム) によるファイル転送を使用し，リアルタイム性の高い処理を実現する．
2) 将来に備え，転送ソフトウェアに依存しない作りにする．
3) ハードウェア，ネットワークの構成を選ばない作りにする．

4.3 結　果（成果物）

4.3.1 転送要求の受付と実転送の非同期な実行

アプリケーションが，リアルタイムにファイル転送を行ったならば，通信ソフトウェアがファイルの送出に成功した時点で，転送完了のステータスが返されるが，転送先の状況によっては，受信処理が遅れたり，送信側では検知できない障害が受信処理側で発生する等により転送処理が遅れた場合，アプリケーションを使用するユーザも次の処理へ進むことができず待ち状態が発生する結果となる．FTS の転送手順では，アプリケーションが転送ファイルを作成後，FTS への転送要求として次のような転送

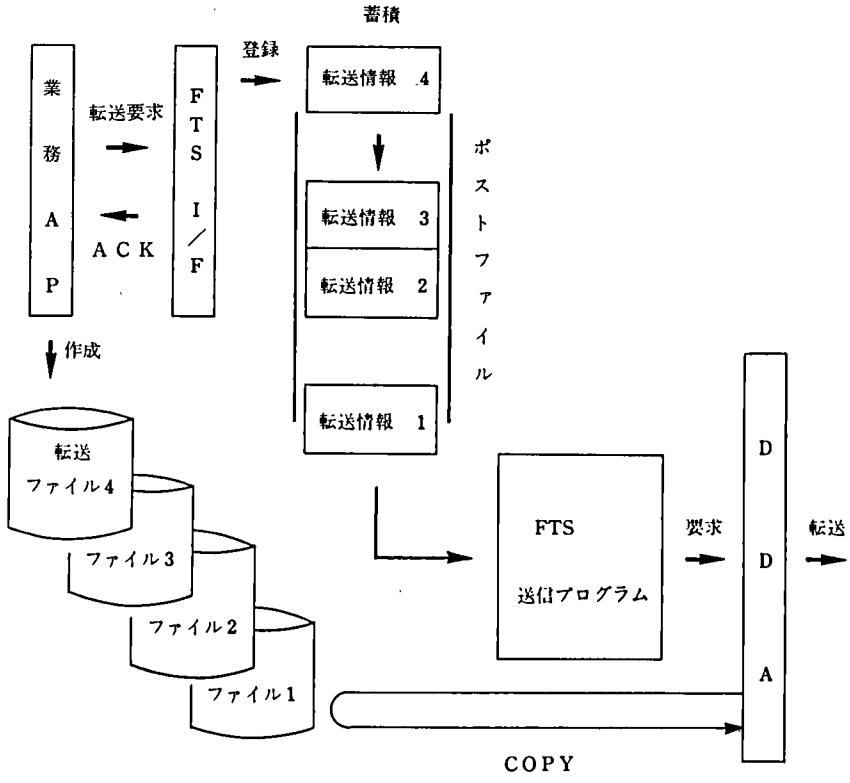


図 4　待ち行列型トランザクション処理

情報がわたされる.

- 転送種別（通常転送，同報転送等の転送種類）
- 転送先ノード名
- 転送元ノード名
- ファイル名
- システムコード（転送先システムのコード）
- データキー（転送先での処理単位）
- データ種別（転送先での処理種類）

この転送情報に対しFTSインタフェースは,”ファイルの有無”,”転送先”をチェック後，転送情報を一旦ポストファイルと呼ばれる蓄積ファイル(図4)に登録し，それと同時にアプリケーションにACKを返すため，転送先の状況に影響されることなくアプリケーションは即時に次の処理に進むことが可能である.

ポストファイルに入れられた転送情報は，転送先が受信処理可能な状況であれば先入れ先出しの処理を守り，FTSの送信プロセスがDDAに対する転送要求を出す間隔と転送間隔をできる限り縮めることにより待ち行列型トランザクション処理を行う.

4.3.2　転送指示順の保障

新営業システムにおいて，業務データの発生順とファイル転送後の受信側アプリケーションの処理順とは同じでなければならない.

たとえば部門サーバで，ある取引に対して新規登録，更新，削除という処理が順に

行われた場合，データの転送先でもこの順序で処理されなければ，登録処理以前に更新・削除処理が行われたり，削除処理後に更新・登録処理が行われる等により，アプリケーションの障害，あるいは転送元と転送先でのデータ不整合が発生してしまう．

このようなトラブルを未然に防ぐには，アプリケーションからの転送要求順にDDA に転送を行わせただけでは解決できない．大容量データの転送中に小容量データの転送を行ったならば，小容量データが先に転送完了してしまい転送の追い越しが発生する可能性があるためである．このため FTS では一つのノードに関して並行転送処理は行わず１ファイルずつの転送を行い，転送先でファイル受信後即時に受信OK の電文を転送元へ返すことにより次ファイルの転送処理に移る．また，転送ファイルは振り付けられた番号により通番管理されているために，もし転送の追い越しが発生した場合でもそれを検知でき，障害として通知する．

4.3.3 転送先非稼働時の転送指示

FTS では他ノードの FTS 稼働状況を自ノードで管理することにより，ポストファイルに蓄積されたアプリケーションからの転送要求を即時にファイル転送として実行するか，あるいは転送要求のみ受け付け実転送は行わないかの制御を行う．そのため各ノードは FTS の立ち上げ時にファイル転送処理対象にしているノードに対して通知を行う．受信側ではそのノードがファイル転送可能状態になったことを検知することができる．また FTS の終了時に相手ノードに終了通知を行い，自ノードがファイル転送が不可状態になったことを通知して，ファイルの転送を停止させる．ファイル転送処理中に相手ノードが停止した場合，転送元では OK 電文を受信できず転送が中断してしまう．そのため相手ノードの立ち上げ通知を再び受信するか，自ノードの FTS が再立ち上げされれば，その中断されたデータは再送される．

このように他ノードの稼働状況管理下で転送処理は制御されているが，アプリケーションの転送要求はこれによることなくポストファイルへ蓄積されていくため，転送先の稼働・非稼働に関係なく転送指示が可能である．

4.3.4 業務アプリケーション制御機能

メインフレームと部門サーバのシステムを連携させることによりデータ交換を行ったならば，ファイル転送終了後に受信側で後処理につなげるために，アプリケーションの起動処理が欠かせなくなってくる．一連の作業に人手を介することなく自動的に行うためには，FTS でファイル転送終了後に，アプリケーションを起動し，転送ファイルを引き渡す処理が必要となる．新営業システムでは，データの種類により起動しなければならないアプリケーションが全く異なり，処理順，起動のタイミングを制御しなければならない．

転送指示，アプリケーション起動のそれぞれの順序が守られたとしても，アプリケーションの処理時間によっては，データ順が前後する．そのため FTS ではデータキーというアプリケーションの処理単位を設定し，そのデータキーが同じデータに関しては同時にアプリケーションが処理されることはなく，前のデータの処理終了後に次のデータに関しアプリケーションを起動させることによってデータの処理制御を行っている．これを実現するために，FTS によりスケジュールされたアプリケーションは，起動時に FTS インタフェースによって開始宣言を行い，ステータスを処理中に更新

し，他アプリケーションのスケジュールを抑止する．処理終了時の終了宣言でステータスを処理済に更新し，次のアプリケーションがスケジュール可能となる．

また，起動されたアプリケーションがエラー終了し，終了宣言を出さなかった場合は，それ以降のデータへ処理が進まないが，障害対応後アプリケーションに対し再起動指示を行い，正常終了させることにより対応が可能とした．

アプリケーションを起動する場合，起動するアプリケーションは転送されてきたデータによりそれぞれ異なる．そのスケジュール方法として，受信側のノードでは条件テーブルと呼ばれる起動アプリケーション制御テーブルをあらかじめ作成する．これは受信したデータに対するアプリケーションの起動情報であり，データに対するデータ種別，アプリケーション名，運用時間帯，起動ステータス（起動可・不可の指示）が登録され，データファイル以外にDDAのオプションとして転送された転送情報(4.3.1項参照）のデータ種別を条件テーブルから検索し該当アプリケーションをスケジュールできる．この条件テーブルにより，業務内容に応じたアプリケーション運用が可能となった．

また各部門サーバでは，朝メインフレームからダウンロードされるコードマスタのデータが反映されるまでユーザによるシステム使用を禁止し，転送されてきたコードマスタ以外のデータに対してもデータベースの更新を行わないようにアプリケーションを制御しなければならない状態が発生する．この場合，前日に運用管理サーバから業務開始前処理を全サーバへ指示し，条件テーブルの起動ステータスをコードマスタおよび制御データ以外のアプリケーションに関し，非稼働にしておく．コードマスタの直後にダウンロード・処理される制御データによる処理で条件テーブルの起動ステータスを反転させ，他アプリケーションを起動可状態にする．

4.3.5 障害検出と通知

システムにおける障害は，他システム同様FTSでも避けることはできない．しかし，1メインフレーム/26サーバの大規模な分散システム全てを人手で監視することは非現実的である．そのためFTSでは，システム内で検出された障害を全てエラーログファイルに記録すると同時に，部門サーバでは運用管理システムの障害通知機能を使用して，障害内容を運用管理サーバへ通知し，通知された運用管理サーバが鳴らす警告音によりシステム担当が検知・対応することを可能とした．また部門サーバでは，それがファイル転送中，つまり転送ソフトウェアあるいは回線等の障害であっても，DDAの障害通知として作成されるメールを常に監視することにより同様に運用管理サーバへ通知することができる．

4.3.6 コード変換

メインフレームと部門サーバ間でファイル転送する場合，必要となるのがコード変換である．FTSでは，EUC(Extended Unix Code)→UNISYS，UNISYS→EUCへの変換を全てメインフレーム上で行うことを前提としコード変換機能の開発を行った．これは，処理能力の劣る部門サーバ上でコード変換を行うことにより，部門サーバ全体のパフォーマンスを低下させるよりも，処理能力の高いメインフレーム上で行う方が効率的であるからである．また，メインフレームから部門サーバへ日次で同報転送によってダウンロードされる多量のコードマスタ関連ファイルは，メインフレー

ムではそれぞれ1ファイルであるが，各部門サーバ上で行えば部門サーバ数だけコード変換を行わなければならず，メインフレーム上で行えば一度のコード変換により全サーバへのファイル転送が可能だからである．

コード変換は，変換されたファイルをアプリケーションへ引き渡すのではなく，アプリケーションによって読み込まれた転送ファイルのレコード単位に，コード変換インタフェースを呼び出すことにより，データ内容に対応した変換を行うことができる．部門サーバからメインフレームへファイル転送されるデータは，ファイルを作成する時に，レコードの変換パターンを表す”変換キー”をレコードの先頭に付加し作成する．変換キーは，数字であるため変換の必要がなく，メインフレームでも部門サーバと同様なコードで読むことができる．メインフレームで，このレコードを読み込み，変換キーとレコードデータをパケットにセットして，コード変換インタフェース（図5)を呼び出したならば，前もって登録してある変換パラメータファイルから変換キーで変換パターンを検索しそのパターンにあったコード変換を行う．

メインフレームから部門サーバへのファイル転送では，コード変換インタフェース

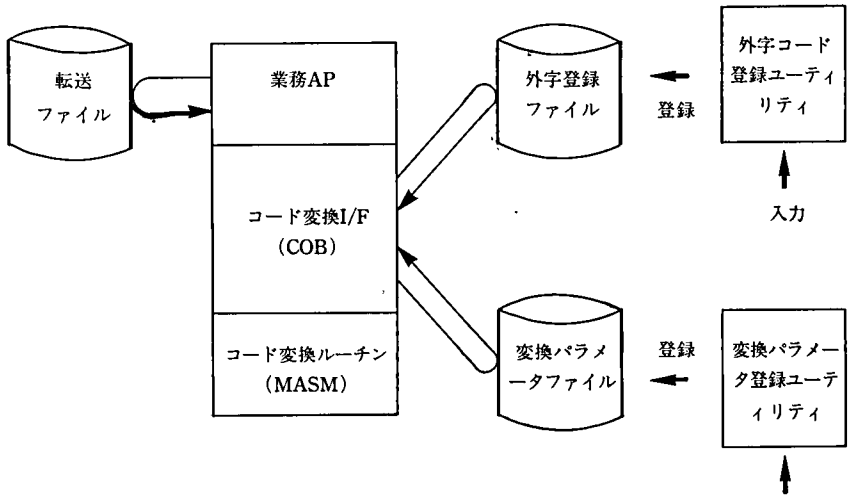


図 5　コード変換概念図

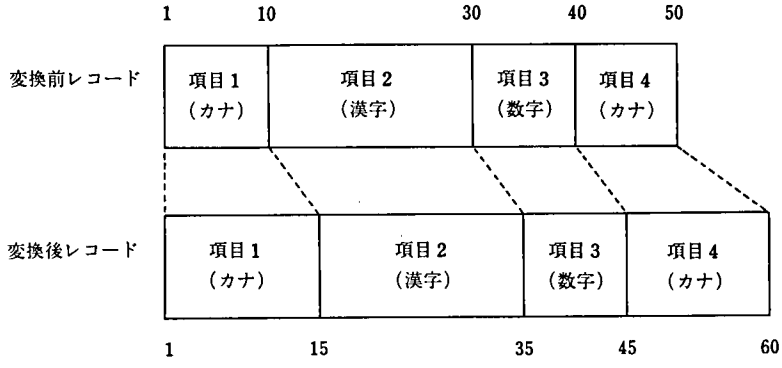


図 6　レイアウト変換

によって変換されたレコードにより転送ファイルを作成する．またFTSでは，単純なコード変換を行ったのでは，レコード内の項目に桁ズレが発生する可能性があり，金額項目等での桁ズレは致命的なトラブルを引き起こしかねないため，一つのレコードの中の項目ごとにコード変換と同時に，レイアウト変換を行うことにより，業務データに適した変換ができるようにした（図6）．

5. 印　書（PSS）

5.1 要　　求

利用者からの印書に対する要求は以下のとおりであった．

1) 通常時はワークステーションに直結している印書装置に印書する．印書装置が障害中のときは，他のワークステーションの印書装置を指定して印書する．
2) 印書中止，印書取消，再印書，印書優先度の変更の指示ができること．
3) 白紙への印書の時には，印書装置があいていれば自動的に印書すること．プリプリントされた定型用紙への印書の時には，用紙セットのタイミングをランプの点灯等で利用者に知らせること．
4) 簡単なプログラミングでオーバーレイ付き印書等の高度な印書が実現できること．

5.2 要求分析と目標設定

以上の要求は，事務処理システムの印書機能としては，どれも当然のものばかりであり，オフィス・コンピュータやメインフレームが実現してきた端末での印書環境そのものであるともいえる．UNIXは印書サービスとしてスプーリング機能を提供するコマンド群を用意しているのでそれらを活用すれば，要求1）と要求2）は容易に実現することができる．

しかし，要求3）の後半部分と要求4）の

・用紙のセットのタイミングをランプの点灯等で使用者に知らせる
・オーバーレイ付き印書

は，UNIXの印書機能では実現がむずかしい．

UNIX印書サービスは，印書データを一旦スプールディレクトリに蓄積した時点で使用者に実行終了の通知を返す．実際の印書は，プリントデーモンが蓄積された印書データを取り上げ，プリンタの排他制御を行いながら印書装置にわたすことで行われる．すなわち，印書の要求と印書装置での印書が非同期に実行される．このため，用紙セットのタイミングを使用者に知らせるのは難しい．プリントデーモンから利用者にメールを送ることは設定により可能であるが，新営業システムはMotif* を使ったGUIを実現しているので，メールによる通知は不適切である．印書制御装置特有の機能を制御する独自機能を作り込めば，印書装置を細かく制御することもできるが，それでは他の印書装置との互換性が無くなってしまう．

オーバーレイ付き印書も，オーバーレイのコードが印書装置ごとに違うし，UNIX印書サービスには印書イメージと合ったオーバーレイを自動的に印書装置に送り込む機能が無いため，UNIX印書サービスでは実現がむずかしい．

---

* Motif は Open Software foundation の登録商標である．

要求項目には明示されていないが，考えなければならない問題としてペーパセービングおよび印書装置の共有化の問題がある．

以上のことを考慮して，次のように目標を設定した．

1) LAN 内の印書装置の共有化を図る．ただし，利用者にはワークステーションに直結している印書装置と同様に使えるようにする．
2) 簡単な操作で印書キューの操作ができるようにする．
3) 手差し印書の時は，用紙セットのタイミングを検出し，印書を指示したワークステーションに Motif の画面で通知する．操作性は，ワードプロセッサの印書機能と同等を目指す．特定の印書装置に依存しない機能にする．
4) オーバーレイと印書データとを，別々に扱えるようにする．アプリケーション・プログラムは印書データのみを作れば良く，オーバーレイと印書データとの併合および印書装置へのオーバーレイの転送はミドルソフトウェアで行う．
5) 印書イメージの表示・編集・他の印書装置への印書等に備え，再利用しやすい形式で印書イメージを保存する．

5.3 PSS（帳票出力支援システム）

5.3.1 分散プリントの全面的採用

帳票出力支援は，ミドルソフトウェア・プロダクトである DstPRT（Distributed Print :分散プリント）で実現した（図 7）．帳票データの作成は，クライアントであるワークステーション上で行う．帳票データの保存と印書指示は，分散プリントサーバで行い，印書はデバイスサーバに接続しているレーザプリンタ（JPU 1028）とインパ

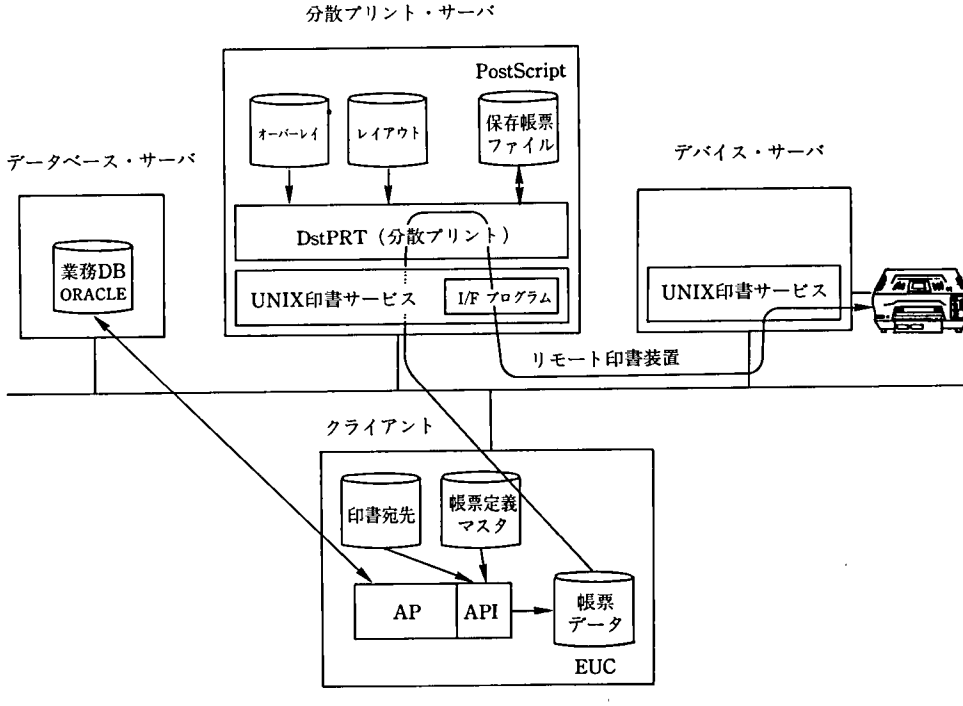


図 7 帳票出力支援システムの構造

クトプリンタ（JPU 0013）で行う．

5.3.2 印書宛先管理

印書宛先の管理は，ワークステーション上の環境変数で行っている．環境変数にはワークステーションごとの印書環境，つまりどこの分散プリントサーバを使ってどこの印書装置に印書するのかを，あらかじめ設定しておく．(通常は，物理的に最も近いプリントサーバと印書装置を設定する.)アプリケーションプログラムからの印書宛先の指定は，直接印書宛先を指示するのではなく環境変数を用いて間接的に行う．これにより，論理的な印書宛先の指定と物理的な印書宛先の管理とを互いに独立して行うことができる．

印書宛先とデバイス・サーバ上の印書装置との関連付けは，UNIX 印書サービスを使って実現している．デバイス・サーバ上の印書装置を分散プリントサーバのリモート印書装置として設定する．

5.3.3 プリンタ操作

UNIX 印書サービスは，印書キューを操作するための十分なコマンド群を提供している．しかし，日頃 UNIX を使い慣れていない新営業システム利用者には UNIX のコマンド群を使いこなすのは難しいため，プリンタのキューの操作は，分散プリントのプリンタ制御機能で実現した．

5.3.4 手差し印書

手差し印書は，UNIX 印書サービスのインタフェースプログラムのカスタマイズと独自に作成した Motif 画面とで実現した（図 8）．

利用者には，以下の二つの画面が通知される．

- 他の利用者が印書装置を使用中であることを通知する画面（図 8 中の画面 1）．
- 印書装置への用紙セットを促す画面（図 8 中の画面 2）．

両画面とも，処理を続けるか/中止するかの選択ボタンを持っている．

画面 1 の起動は，API が印書要求を UNIX 印書サービスにわたす時点で行う．印書

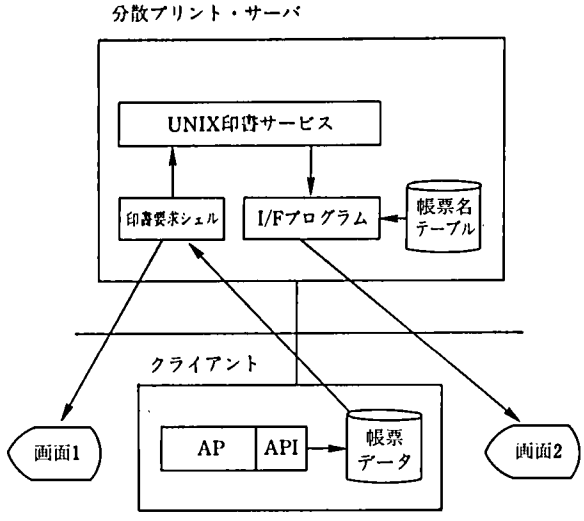


図 8　手差し印書メッセージ

要求シェルは指定された印書宛先のキューの有無を調べ，キューがあれば画面1を起動する．この時利用者に通知される情報は以下の二つである．

・印書装置が空くのを待っている人の数．

・どのワークステーションの使用者が待っているのか．

画面2の起動は，プリントデーモンが印書キューを取り上げインタフェースプログラムを起動した時点で行う．インタフェースプログラムは，以下の情報を利用者に通知する．

・帳票ID

・セットすべき用紙名

・印書装置名

利用者は，このメッセージを受け取ることにより，用紙セットのタイミングを知ることができる．

5.3.5 レイアウト・オーバーレイと帳票データを分離

用紙サイズ・用紙の方向・文字の位置および属性などのレイアウト情報とオーバーレイ情報は，帳票のデザイン時に作成し分散プリントサーバ上で管理する．アプリケーションプログラムは，論理データに帳票の属性を付加し帳票データを作成する．帳票の属性とはレイアウト名・オーバーレイ名・印書部数・保存日数・セキュリティ・使用者指定情報などのことである．

5.3.6 PostScript形式での保存

アプリケーションプログラムが作った帳票データは，分散プリントAPIにより分散プリントサーバに転送され，帳票属性に従ってレイアウト・オーバーレイとマージされる．マージ後の帳票データは，PostScript* 形式に変換され，保存・印書される．PostScriptは，特定のハードウェア・ベンダに依存しないページ記述言語であり，テキストだけでなく図形の表現にも優れている．このため，以下のようなメリットを期待できる．

・業界標準であるため，印書装置の多様化に対応しやすい．

・印書イメージを画面上に表示することができる．

・言語としての表現力が豊富なので，将来の帳票のカラー化にも対応できる．

6. メニューシステム（MNS）

6.1 要　　求

利用者側からのメニューシステムに対する要求は以下のとおりであった．

1) ワークステーションの画面で分散システムのアプリケーションとメインフレームのアプリケーションとを自由に使えること．

2) 利用者ごとに，使えるアプリケーションや機能が制限できること．

6.2 要求分析と目標設定

要求1)を実現するためには，実行方法の異なる以下の3種類のアプリケーションの実行を可能にしなければならない．

・ローカル（クライアント）で稼働するアプリケーション

---

* Post Scriptは米国Adobe Systems社の登録商標である．

・リモートのホストまたはワークステーション上で稼働するアプリケーション
・メインフレーム上で稼働するアプリケーション

接続方法，実行環境設定方法，使用するソフトウェアは三者三様である．また，アプリケーションによっては，ファンクションキーに特別な意味付けをしたり，特殊な印書操作を行うものもあるので，その点も考慮する必要がある．

UNIX でのセキュリティは，パスワードファイルによるログイン時の正当性チェックとファイルのパーミッションによる実行許可制限とで実現する．ログイン名ごとに実行許可と実行権限の制限を行うのを基本としているのである．要求 2) を満たすためには，営業マンごとにログイン名を決め全国のサーバ・ワークステーションに設定すればよい．しかし，全国に約 1000 人いる営業マンのログイン名を全国のサーバ・ワークステーションに設定し，その後も維持していくのは容易なことではない．また，UNIX の機能だけではアプリケーションレベルの細かい実行許可制限は実現できない．したがって，UNIX でのセキュリティ以外の独自のセキュリティ機能を実現する必要がある．

以上のことを考慮して，次のように目標を設定した．

1) 実行プラットフォームへの接続，実行環境設定，およびミドルソフトウェアへの実行時パラメタの設定は，全てメニューシステムで行う．
2) UNIX のセキュリティとは別なセキュリティ機能をつくり，UNIX の機能と組み合わせて，セキュリティチェックと実行許可制限を実現する．

6.3 MNS（メニューシステム）

6.3.1 ユーザ管理ファイルによるセキュリティチェック

ユーザごとの実行許可制限を管理するユーザ管理ファイルを作り，セキュリティチェックと実行許可制限を実現した（図 9）．

ユーザ管理ファイルには以下の情報をあらかじめ登録しておく．

・ユーザ ID
・パスワード
・氏名コード

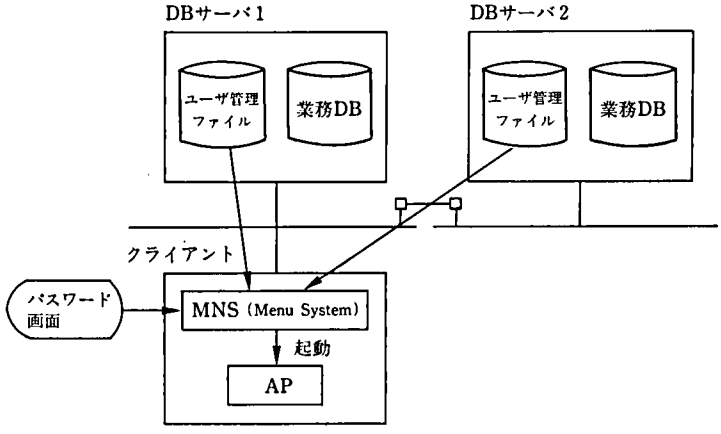


図 9　統合メニューの構造

・セキュリティコード

セキュリティのチェック，すなわち正当なアクセス権を持ったユーザかどうかのチェックは，次の3段階で行っている．

1) 基盤システムを使えるユーザID/パスワードを使ってワークステーションにログインしたか．

2) ユーザ管理ファイルに登録済みのユーザIDでMNSにログインしたか．

3) 指定したアプリケーションを実行できる実行許可レベルが設定してあるか．

実行許可レベルは，ユーザ管理ファイルのセキュリティコードにユーザIDごとに設定しておく．MNSでは，10種類のカテゴリ別にそれぞれ実行許可のレベルを登録できるようにした．カテゴリごとの実行許可レベルをどのように扱うかは，アプリケーションにまかせている．

6.3.2 アプリケーションの起動

上記のセキュリティチェックをクリアした場合のみ，アプリケーションを起動する．この時，以下の情報を起動時引き渡し情報としてMNSからアプリケーションプログラムに引き渡す．

・ユーザID
・パスワード
・氏名コード
・セキュリティコード
・ORACLEサーバID
・FTS（ファイル転送）サーバID

基盤システムとアプリケーションシステムの実行環境の設定は，UNIX環境変数を使って行っている．ログインシェルで基盤システムとアプリケーションシステムの環境変数設定用シェルを起動するのである．環境変数の中には，ワークステーションごとに固定的に決まるものと，実行時に動的に決まるものとがある．

ワークステーションごとに固定的にきまる情報には以下のものがある．

・PSS（印書）サーバID
・部門サーバID

実行時に動的に決まる情報には以下のものがある．

・ORACLEサーバID
・FTSサーバID

6.3.3 メインフレーム端末のエミュレーション

メインフレーム端末のエミュレーションは，IS 6000 (Information Service) の会話型サービスを使って実現した（図10）．IS 6000は部門サーバ上にあるので，ワークステーションからはリモートログインしてIS 6000のサービスを利用するようにした．

メインフレーム上のアプリケーションを使用するためには，通常以下のような手順を利用者が会話処理で行わなければならない．

1) IS 6000会話処理を起動し，目的のアプリケーションが存在するメインフレームとの接続を確立する．

2) メインフレームにサインオンする．（ユーザID/パスワードを入力する．）

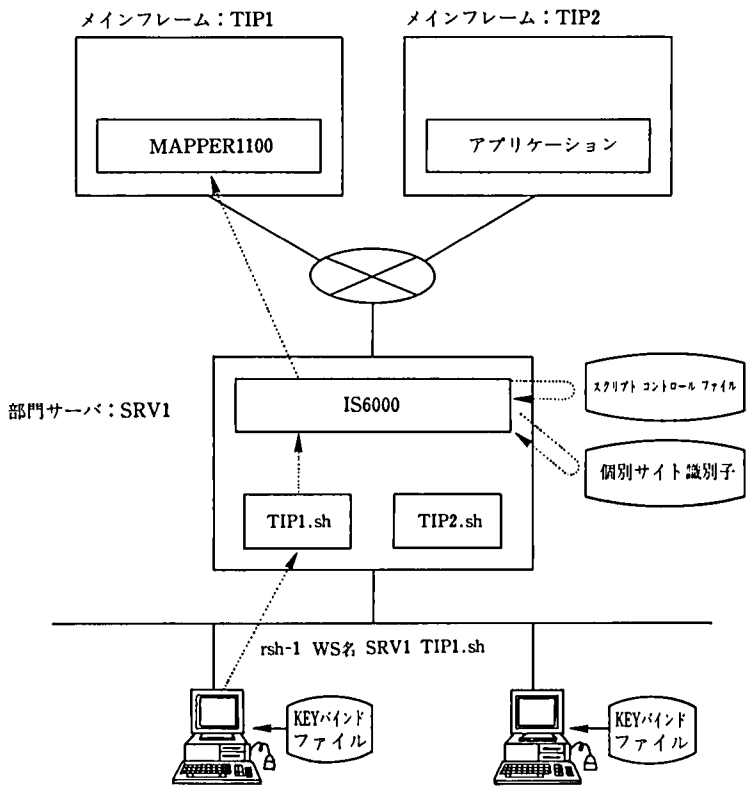


図 10　メインフレーム端末エミュレーション

3)　アプリケーションを起動する.

4)　アプリケーションにサインオンする.（ユーザ ID/パスワードを入力する.）

MNS では，IS 6000 の ISSC（IS スクリプトコントロール機能）を使って，上記手順 1）～3）までの自動化を実現した．メインフレームとの会話処理は，スクリプトファイルと呼ばれるパラメタファイルにあらかじめ記述しておき，IS 6000 の会話型サービス起動時に引き渡す．ISSC を使えば，上記の全ての手順を自動化することも可能であるが，セキュリティを考慮して，手順 4)の自動化は止めた．利用者は，ワークステーション上に用意されたアイコンをクリックするだけで，メインフレームのアプリケーションを使用することができる.

アプリケーション特有のキー設定は，KEY バインドファイルに設定した．KEY バインドファイルは，ワークステーション上に端末画面が開かれる時に参照されるファイルである.

アプリケーションの中には，4 GL の MAPPER のように端末識別子や PID(Position ID)から端末を特定し，常に決まった端末へ印書イメージやメッセージを送付するものがある．IS 6000 による会話型処理では，通常実行ごとに割り振られる端末識別子が異なるが，MNS では IS 6000 の個別サイト識別子機能を使って，ワークステーションと端末識別子との対応を固定することでこの問題を解決した．個別サイト識別子機能は，使用者識別名とサイト識別名の関係を設定できる機能である．MNS では，ワ

ークステーションIDを使用者識別子として個別サイト識別子機能に登録している.

7. ネットワーク支援 (NSS)

7.1 要　　求

利用者からのネットワーク支援に対する要求は以下のとおりであった.

1) UNIXサーバとUNIXワークステーションとでクライアント/サーバ処理を行う.
2) 全国に分散配置されている26台のUNIXサーバとサーバ配下の130台のワークステーションの環境, つまりUNIXネットワークファイルや実行モジュールなどを常に同一に保つ.
3) ただし, サーバの運用時間は各営業部門に独自に決めさせる.
4) UNIXワークステーションの運用時間も, 利用者に独自に決めさせる.
5) 営業部門内のUNIXサーバとUNIXワークステーションとで, ファイルの共有と印書装置の共有を実現する.
6) APプログラムでは, プラットフォームの違いを意識しなくて良い.

7.2 要求分析と目標設定

UNIX同士のクライアント/サーバ処理は, UNIXのネットワーク機能を利用すれば容易に実現することができる. リモート側のリソースをアクセスする手段は, Rコマンドやソケットライブラリが提供してくれるので, 利用者はUNIXネットワークファイルの設定をするだけでよい. とくにネットワークOSを必要としないのは, UNIX同士のクライアント/サーバ処理の利点の一つである.

しかし, 全国各地に分散配置されたUNIXサーバとUNIXワークステーションのUNIXネットワークファイルを整合性を維持しながら同期をとって更新していくのは容易なことではない. 同期をとって更新していかなければならないものには, 以下のものがある.

- UNIXネットワークファイル (hosts, services等)
- 基盤システムやアプリケーションシステムの実行モジュール
- 基盤システムやアプリケーションシステムのシステムファイル
- 各マシンのシステムタイム

各マシンがバラバラな運用時間をとるようなシステムでは, 変更情報を同期をとって配布するのはむずかしい. 配布時にマシンが停止中のこともあるし, マシン同士で業務日付けが違うこともありうる. すべての場合を想定して配布を考えると, 配布システムがとても複雑なものになってしまう. そこで今回は, 配布ではなく集信方式をとることにした. 集信方式とは, クライアント側が主体的にサーバ側へ情報を採りにいく方式のことであり, 配布方式に比べて次のような長所を持っている.

- サーバ側はクライアント側の詳細な状況を全て把握していなくても良い.
- 配布漏れが無い.
- クライアント側の都合で勝手に運用できる.

以上のことを考慮して, 次のように目標を設定した.

1) 全社的な環境は, マスタ・サーバで集中的に管理する. 全国のサーバ/ワークス

テーションへは，集信方式で整合性を維持しながら同期をとって反映する．

2) ファイルや印書装置の共有などの部門独自の環境は，部門サーバで集中的に管理する．配下のワークステーションへは，集信方式で部門独自の環境を反映する．

7.3 結 果（成果物）

7.3.1 NSS ドメイン

上記要求を実現するにあたり，NSS ではインタネット・ドメインに準拠したネットワーク階層を想定した（図 11）．

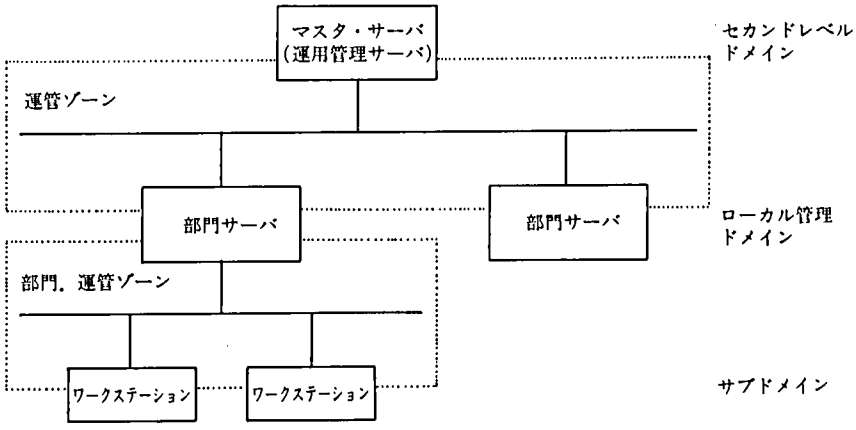


図 11　NSS ドメイン

運用管理サーバをセカンドレベルドメイン，部門サーバをローカル管理ドメイン，ワークステーションをサブドメインと位置付けた．NIC (Network Information Center) には加入していないが，将来の加入を考慮した．

各ドメインは，ホストの階級的な共同体であるゾーンで区切られ，一つの権限によって管理される．NSS では，運用管理サーバと部門サーバを運管ゾーンとし，運用管理サーバがゾーンを管理するとした．同様に，部門サーバと配下のワークステーションを部門・運管ゾーンとし，部門サーバがゾーンを管理するとした．ゾーン内の情報は，全てゾーンを管理するサーバが管理する．ゾーン内のドメインは，必要な都度ゾーンを管理するサーバに問い合わせる．

今回のシステムは，運用管理サーバを頂点とする 3 階層のシステムであるが，NSS は，多階層のシステムでも同様の振る舞いをするようにした．

7.3.2 リプリケータサービス

リプリケータサービス (図 12) は，NSS マスタである運用管理サーバのオブジェクトを，自動的に配下の部門サーバ/ワークステーションに配布する機能である．

オブジェクトとは配布対象物のことであり，

- 実行プログラム
- データファイル
- 環境変数設定シェル

などのことである．リプリケータサービスでは，配布対象物と配布情報とを別々に管理している．配布対象物は，配布先と同じデイレクトリに管理している．いつ何を配

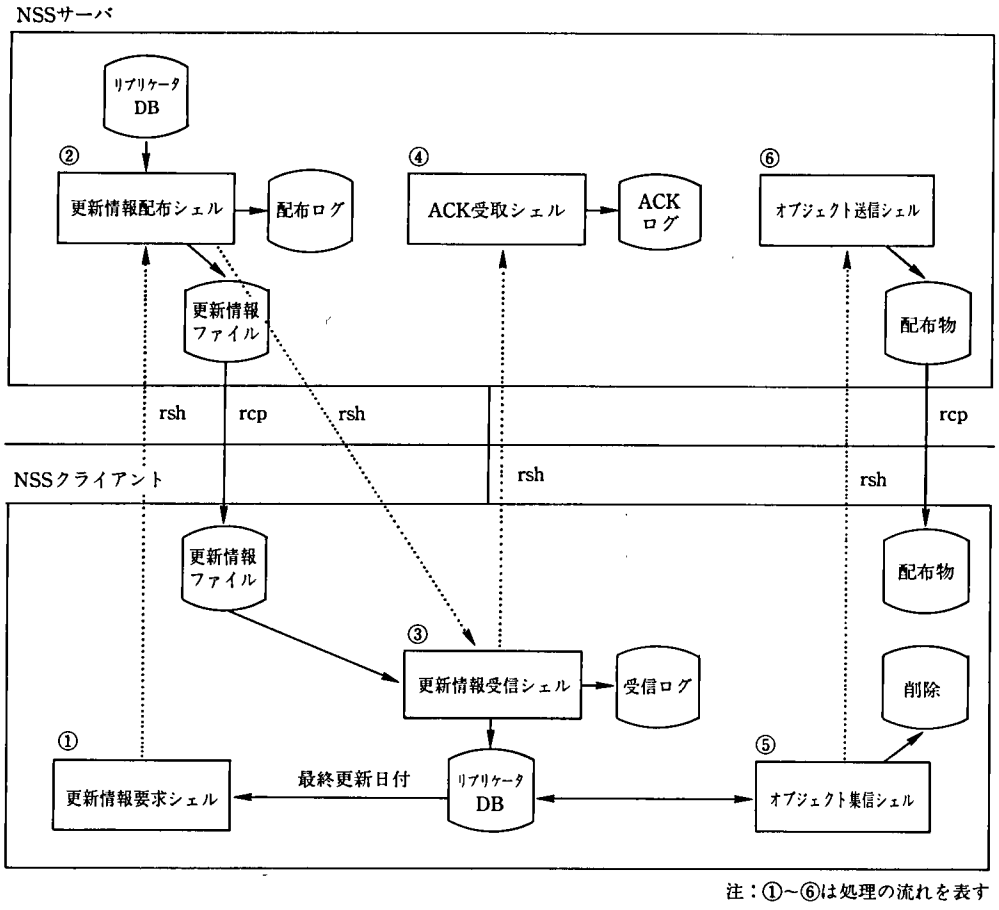


図 12 NSS リプリケータ

布するかといった配布情報は，リプリケータ・データベースという UNIX ファイルに配布実施日付けごとに管理している．

実際の配布は，UNIX の rcp コマンドを使った集信方式で行っている．処理の起動は，NSS クライアントの立ち上げ処理時にクライアント側からサーバへオブジェクトの転送要求をすることで行う（図 12 の①）．このときクライアントは，自サーバのリプリケータデータベースの最終更新日付けをサーバへ引き渡す．転送要求を受けた NSS サーバは，クライアント側の最終更新日付けと自サーバのリプリケータデータベースの最終更新日付けとを比較し，送るべきオブジェクトを決める（図 12 の②）．立ちあげ処理時にクライアント側から集信することにより，クライアント側の事情で数日間クライアントマシンが稼働していなくても，次に立ち上がる時には，最新のオブジェクトを反映して立ち上がることができる．転送するオブジェクトは，

- 転送の効率化をはかる
- オブジェクトの所有権モードを変えずに送る
- オブジェクト作成日付けを変えずに送る

ために，cpio コマンドで複数オブジェクトを一つにまとめ，compress コマンドで圧縮

して送り，クライアント側で uncompress コマンドを使って復元するようにしている．

NSS のリプリケータサービスは，クライアント側からの集信を運用管理サーバ/部門サーバ/ワークステーションとで階層的に繰り返すのを基本としている．配布されるオブジェクトは，運用管理サーバ，部門サーバ，ワークステーションと順次転送されていく．これは，

- 運用管理サーバは 24 時間稼働している
- 部門サーバの運用時間は，部門ごとにまちまちであるが，一日一回はシャットダウンする

ことを前提としている．ところがワークステーションの立ち上げ，シャットダウンはユーザにまかされているため，必ずしも毎日確実にシャットダウンが行われるとは限らない．シャットダウンが行われないと NSS が起動しない．そのためそのワークステーションは，UNIX ネットワークファイルの更新や，オブジェクトの配布を受けないまま稼働し続けることになる．このような事態を避けるために，NSS は部門サーバ上で NSS クライアントが起動した時に，部門サーバ配下の全てのワークステーションをリモートから強制的に終了させるようにした．

7.3.3 ネットワーク環境設定

ネットワーク環境設定機能は，NSS マスタである運用管理サーバの UNIX ネットワークファイルに加えた全ての変更を自動的に配下の部門サーバ/ワークステーションに反映させるための機能である（図 13）．

UNIX ネットワークファイルに加えた変更は，変更箇所単位にネットワークデータベースという名前の UNIX ファイルで管理する．

集信のしかたは，前述のリプリケータサービスと同様である．

7.3.4 共通時間設定

共通時間の設定は，UNIX の rdate コマンドを使って実現した．rdate コマンドは，リモートシステムのシステムタイムをローカルのシステムタイムとして設定するためのコマンドである．基準となる時間は，運用管理サーバのシステムタイムである．各部門のサーバは，立ち上げ処理時に運用管理サーバに rdate コマンドを発行し，ローカルタイムを運用管理サーバの時間に合わせる．部門のワークステーションは，立ち上げ時に部門サーバに rdate コマンドを発行し，ローカルタイムを部門サーバの時間に合わせる．

厳密には，ローカルとリモートの間の通信の時間分の誤差が発生するが，ms 単位の時間を要求するシステムではないので実害はない．

7.3.5 リモートファイルアクセス

リモートファイルアクセスは，NFS（Network File System：分散型ファイルシステム）を利用して実現するのが簡単である．しかし，新営業システムでは広域網を含む複数ネットワークを経由してのクライアント/サーバ処理があり，下位のプロトコルに非接続型の UDP（User Datagram Protocol）を使っている NFS では，安定性に不安があった．そこで接続型の TCP（Transmission Control Protocol）を使った汎用ファイルアクセスルーチン RFCS（Remote File Control System）を作り，すべてのリモートファイルのアクセスを RFCS を通して行うようにした．

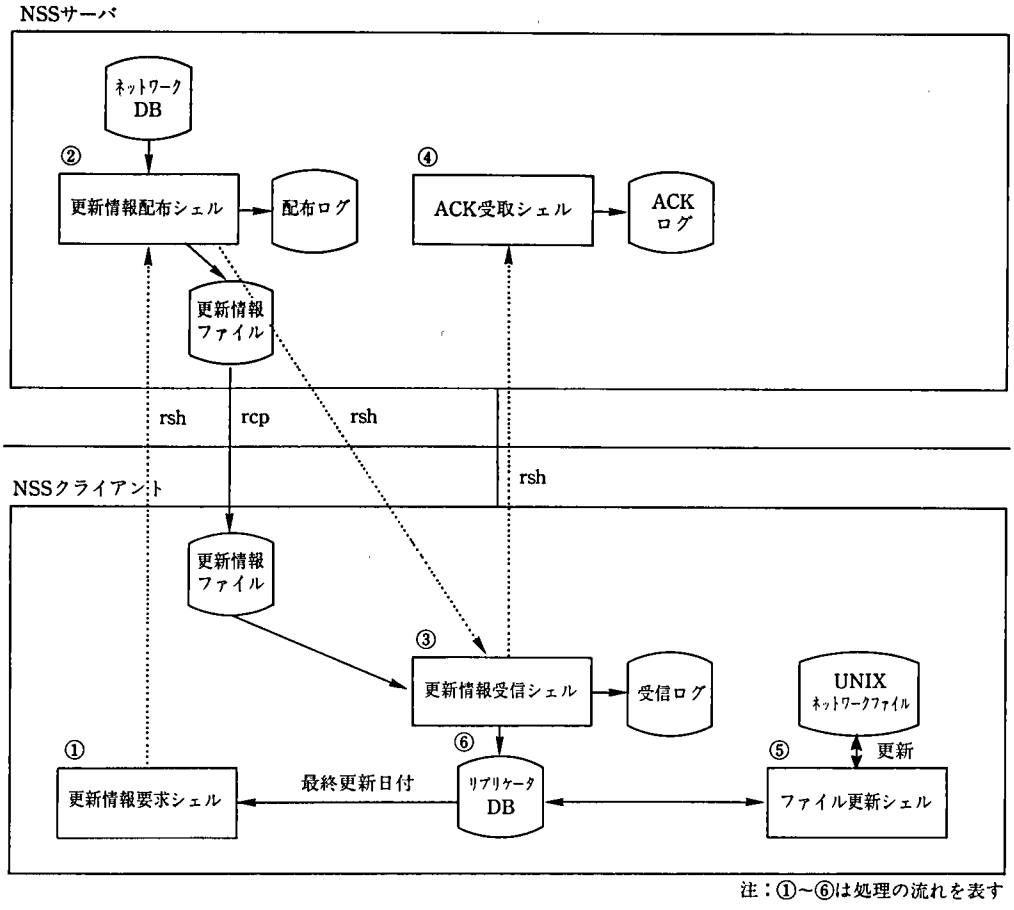


図 13　ネットワーク環境設定

8. 評価・今後の課題

稼働開始から 6 か月を経た現在，基盤システムは 1 日あたりファイル転送約 3000 回，インパクトプリンタへの印書約 1300 枚，レーザプリンタへの印書約 600 枚，オブジェクト配布約 10 件，AP システムの起動約 2000 回の処理をこなしながら順調に稼働している．

DDA によるファイル転送を使った FTS は，1 サーバ 1 件あたり約 30 秒の処理時間でメインフレームと全国 26 箇所の部門サーバ間での非同期のトランザクション処理を実現している．1 件あたり 30 秒という処理時間は決して早いとはいえないが，アプリケーションシステムからの転送要求が一日 100 回以下の部門サーバが多いので問題はない．しかし，決算月等トランザクションが集中して大量に発生する時には，1 件あたり 30 秒の処理時間では遅すぎ，部門サーバ側に転送待ち行列が発生する．また，26 箇所の部門サーバから同時に転送が発生すると，転送指示順を保証するための受信 OK 電文自身がメインフレーム上の DDA 転送待ち行列に入ってしまい，いっそうパフォーマンスを低下させるという現象も出ている．FTS の効率改善を重ね，パフォー

マンスアップを計ってきてはいるが，やはりファイル転送をベースとしたトランザクション処理には自ずと限界がある．FTS は転送ソフトウェアに依存しないように設計してあるので，早い時期に転送ソフトウェアを交換し OLTP の実現を計りたい．

分散プリントを全面的に使った PSS は，アプリケーションシステムからの要求どおりメインフレームの印書環境と同等の印書環境を実現している．オーバーレイ付きの高度な印書を簡単に開発・実行できるということで，アプリケーション開発者や利用者から好評を得ている．印書データをオーバーレイとマージして印書装置固有のコードに変換する処理は，とても CP U 負荷が高い．そのため，アプリケーションシステムのパフォーマンスが低下するという現象が発生したことがある．基盤システムでは，この問題を分散プリントサーバをデータベースサーバと別々にすることで解決した．部門サーバ集中型の基盤システムだったら，部門サーバを能力の高い機種に変更しなければならなかったところである．やはり水平分散型のシステム構成も可能にしておいたのは正しい選択であった．PSS は今回，メインフレームの印書環境をシミュレートすることを目標として開発してきたが，今後は分散システムと分散プリントの特徴を生かして

・帳票イメージをワークステーションの画面に表示し変更する，
・帳票データをパーソナルコンピュータ上のワープロソフトウェアへ引き渡す，
・パーソナルコンピュータのデータを印書データとして取り込む，

を実現していくつもりである．

ユーザ ID とパスワードを入力してアプリケーションシステムを利用するというスタイルはメインフレーム時代からとっていたものなので，MNS は利用者に違和感無く受け入れられている．利用者とアプリケーションシステムは，クライアントであるワークステーションがどのサーバと処理をしているのかを意識することなく処理を実行することができている．

集信方式を採用した NSS のおかげで，部門サーバ/ワークステーションは，メインフレームや他の部門サーバの運用時間にとらわれずに，常に最新のオブジェクトやネットワークファイルを入手して稼働することができている．また，集信方式のため NSS 自信もシンプルな作りにすることができた．

9. おわりに

基盤システムで使った UNIX やミドルソフトウェアの利用技術は，どれもが既知のものでありとりたてて新しいものは無い．個々の利用技術についても，より良い方法が有り得るであろう．しかし，いろいろな利用技術を組み合わせ，全体として大規模な分散システムの基盤システムを構成している点では，この事例も一見の価値はある．この事例が，今後同様のシステムの開発に従事する方の参考になれば幸いである．

執筆者紹介 小　関　　洋 (Hiroshi Koseki)
昭和35年生. 59年上智大学経済学部卒業. 同年日本ユニシス（株）入社. 以後社内システムの開発に従事. 現在情報システム部販売システム室に所属.

芝　尾　文　彰 (Fumiaki Shibao)
昭和37年生. 62年中央大学法学部卒業. 同年日本ユニシス（株）入社. 以後社内システムの開発に従事. 現在情報システム部利用技術課に所属.

UNIXシステムの運用管理

The Operation of a UNIX-based Application System

内池貴明

要　約　従来のメインフレームホストと端末からなる基幹業務システムを，UNIX*ベースの26台のサーバと123台のワークステーションをネットワークで結んだ分散システムに再構築した．各部門には1台から数台のサーバを配置し，それぞれの部門に運用管理者が選任され，基幹業務システムの運用の一部を担うことになった．しかしながら，各部門の運用管理者は経験・知識の十分でない一般の社員が担当することから，運用を支援するための方策が必要になる．とくに障害時における障害の認識から対応処置，復旧作業までの過程を部門運用管理者が担当することは困難である．

本稿は，基幹業務システムをUNIX上に構築する場合に運用面でどのような配慮が必要であったか，その実例を紹介する．

Abstract Most recently, the author joined in the program to reconstruct a key applications system configured with a traditional mainframe host and terminals into a distributed processing system networking 26 servers and 123 workstations in the UNIX environment. With one to several servers installed at each department, an operation administrator was appointed at every site to take care of part of the operation of the new key applications system. However, since ordinary office workers lacking in computer experience and expertise turned into those administrators, the needs for some measures to support the system operation cropped up. It was found particularly difficult for them to properly react to the whole process ranging from trouble warnings, through response, to action for recovery.

Founded on the author's experience, this paper describes what had to be taken into account in terms of system operation in recreating a UNIX-based information system for mission-critical business applications

1. はじめに

営業情報システムはメインフレームと端末による集中処理形態で運用されていたが，全国の各営業部門に複数台のUNIXのサーバとワークステーションを配置する分散処理形態を採用する新システムに変更した．メインフレームホストによる集中処理の運用管理は，専門の運用部隊を組織したり，外部の運用会社に業務委託することが可能であるのに対し，複数のUNIXサーバを用いる分散処理形態では，運用管理専門の要員を配置し，運用管理業務を専任とする組織を各営業部門ごとに置くことは，人的にもコスト的にも適切ではない．新システムでは，運用管理担当者（以降，運用管理者と呼ぶ）はコンピュータシステムの経験・知識のない一般の営業部門のスタッフであることを前提にし，これらの人々が最低限の作業でコンピュータ運用をまかなえるようにするための運用支援機能を提供している．

本稿は，2章で「システム構成と運用支援に必要とされる要件」について述べた後，

* UNIXオペレーティングシステムはUNIX System Laboratories, Inc. が開発し，ライセンスしている．

運用支援機能の目標と実際について各章に分けて記述する．

3章　部門サーバのセットアップ

4章　部門サーバの終了処理

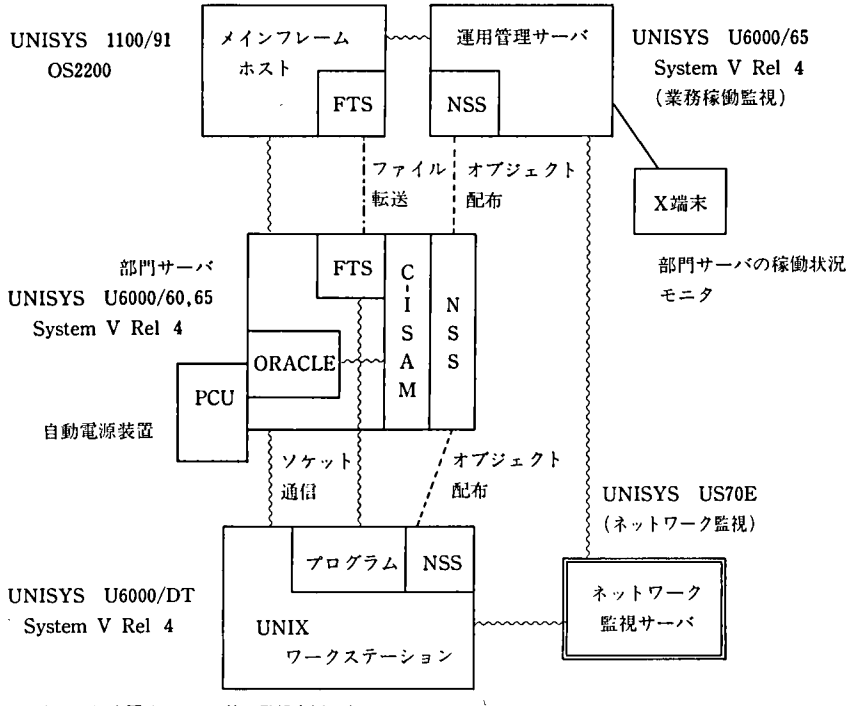
5章　部門サーバの稼働監視

なお，新システムの概要については本誌別稿の「新営業システムの概要—典型的な分散システム」を，また，新システムを支える基盤系ソフトウェアについては「クライアント/サーバ・コンピューティングを指向した新営業基盤システム」を参照されたい．

2. 新システムの構成と運用支援に必要とされる要件

2.1 システム構成と各システムの役割

新営業システムは図1のとおり，メインフレームホスト・マシン，部門サーバ・マシン，ワークステーションの三つのコンピュータ群によって構成される．業務プログラムはワークステーションで保持/実行され，データベースは部門サーバに置かれる．ワークステーションで発生したトランザクションは部門サーバのデータベースに反映され，トランザクションの種類により必要に応じてメインフレームホストに送られる．またメインフレームホストからは，商品コードや客先コードなどのマスタデータ（これをコードマスタと称する）やバックエンドの基幹システムから出力されるデータが各部門サーバに送られる．メインフレームホストと部門サーバ間のトランザクション



・ORACLEは米国ORACLE社の登録商標である．
・C-ISAMは米国インフォミックスソフトウェア社の登録商標である．

図1　システム関連図

を転送する役割を担っているのがFTS（File Transfer System）である．

業務プログラムファイルを部門サーバに保持すれば，プログラムの管理がサーバ単位に行える利点はあるが，プログラムのローディング時間の短縮，ネットワーク上のトラフィック量の軽減などを考慮して，プログラムファイルは各ワークステーションに保持する．また，部門サーバとのプロセス間通信を行わないファイルや業務プログラムのログファイルなども同様にワークステーションに置かれる．プログラムやファイルをワークステーション上に最新の状態で維持するために，部門サーバとワークステーション間でプログラムやファイルの送受信が発生する（以降，これらの送受信対象物をオブジェクトと称し，その転送を単にオブジェクト配布と呼ぶ）．オブジェクトの配布は運用管理サーバから部門サーバに，さらに部門サーバから各ワークステーションに配布される．配布方法は，オブジェクトを欲する側が転送の要求を行う形態をとっており，要求する側を主局，要求に応える側を従局と呼ぶ（主局，従局はクライアント，サーバとそれぞれ読み代えることもできる）．運用管理サーバと部門サーバ間では部門サーバが主局で運用管理サーバが従局，部門サーバとワークステーション間ではワークステーションが主局で部門サーバが従局となる．

運用管理サーバはオブジェクト配布のサーバとしてだけでなく，部門サーバの稼働状況を把握するためにも用いる．原則として部門サーバの運用管理は各部門が行うが，障害時には，障害の認識から対応，復旧作業に至る過程において各部門の運用管理者が対処するのは困難であるので，情報システム部門が障害監視を集中して行い障害時の助言と対応作業の支援を行う．オブジェクト配布を行うためのソフトウェアがNSS（Network Support System）である（図1参照）．

FTS，NSSは基盤システムとして新システムのために作られたソフトウェアであり，本誌別稿の「クライアント/サーバ・コンピューティングを指向した新営業基盤システム」を参照されたい．

ネットワーク監視サーバの目的はネットワークの接続，ネットワークの稼働状況をSNMP（Simple Network Management Protocol）を使って監視することにある．業務稼働監視とネットワーク監視を1台の機械で行うこともできるが，ネットワークの運用は特定業務システムに依存しないため別の組織部門が管理することが望ましく，それぞれ別のサーバで行っている．

2.2 運用支援の要件

新システムの運用は次の三つに大別できる．

- 部門サーバ（とワークステーション）自体の運用
- メインフレームホストと部門サーバの連動運用
- 運用管理サーバと部門サーバの連動運用

1) 部門サーバ（とワークステーション）自体の運用……部門サーバを業務システムの運行に供するための一切の作業，すなわち休日・稼働日の設定から始まって稼働当日の電源投入後のセットアップから終了・電源切断までの機械運用に関わる部門サーバ自体の運用を指す．ほとんどの処理を自動化し運用管理者の作業をできる限り減らすことと，とくに障害時の対応ではUNIXのコマンドやシェルの知識がなくても対応できるような考慮が求められる．また，それぞれの部門で部

門サーバの自主運用を可能にするためには，休日に運用したり早朝からあるいは深夜まで稼働させる場合の運用負荷を軽減することが望まれる．このため，自動電源装置（以下，PCU と呼ぶ）の導入は不可欠となっている．

2) メインフレームホストと部門サーバの連動運用……コードマスタのコード体系が変わってしまうような場合では，その更新途中に業務プログラムが実行されるとそのプログラムはコードの不整合を起こしてしまうため，コードマスタの更新と業務プログラムの実行を同時に行わないような制御が必要になる．

メインフレームホストと部門サーバ間のファイル転送は，操作時点と実際のファイル転送とで多少のタイムラグがあり，また実際の転送はサーバでバックグランド処理されるためワークステーションを操作する利用者は転送上の異常を知ることができない．また，運用管理者は運用管理が専任でないため障害の発生を監視することも期待できない．このため，運用管理者に期待する作業を最低限にすることが必要になる．

3) 運用管理サーバと部門サーバの連動運用……運用管理サーバと部門サーバとの連動はオブジェクト配布時に必要で，従局側が主局側より先に稼働状態になければならない．運用管理サーバは 24 時間無休運転を基本としているので，運用管理サーバと部門サーバとでセットアップが前後し従局側が未稼働状態の場合はない．したがって，主たる運用作業はオブジェクト配布上の障害時だけである．

3. 部門サーバのセットアップ

3.1 セットアップの作業

NSS で配布されるオブジェクトの中身はプログラムやプログラムが使用するファイルであり，利用者がこれらのオブジェクトを使用中に入れ換えて異常動作を引き起こさないようにするために，配布はセットアップ時点で行う．9 時に業務開始するためにはこれらの処理の余裕を見て部門サーバの電源投入を 8 時までには行わなければならない．配布すべきオブジェクトは最大で 8 Mbyte (compress で圧縮後のサイズ) まで取り扱うことができ，これを 64 Kbps の回線で結んだ本社以外の一つのサーバに送るのに約 30 分要する．ただし，9 箇所の支社・支店宛にオブジェクト配布する場合は，

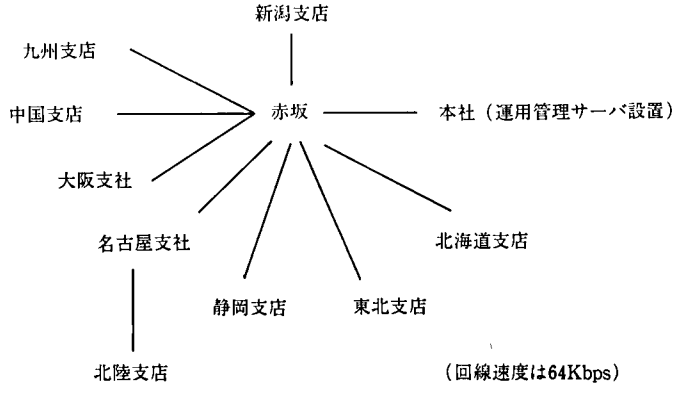


図 2 新営業システムのネットワーク

ネットワーク(図2)上，運用管理サーバが本社に設置されているので，赤坂一本社間の通信がシリアルになり，配布し終わるのに約4時間を要する．実際には圧縮後のサイズで8 Mbyte を送ることはないので，通常4 Mbyte と見積っても支社・支店は午前6時にはサーバの電源投入が必要になる．

しかし，実際に運用管理者が毎日午前6時，7時に出社してサーバの電源投入作業を行うことはできないので，PCU によるセットアップ処理の自動化が計られている．また，PCU と自動電源制御ソフトウェアを採用することにより，電源オン・オフの時間だけでなく日付の指定もできるので稼働日と運休日の設定も可能になる．

3.2 セットアップ処理

UNIX では，init プロセスが参照する inittab にランレベルに応じたプロセスを定義しておくことによって所定のプロセスを起動させることができる．セットアップ処理

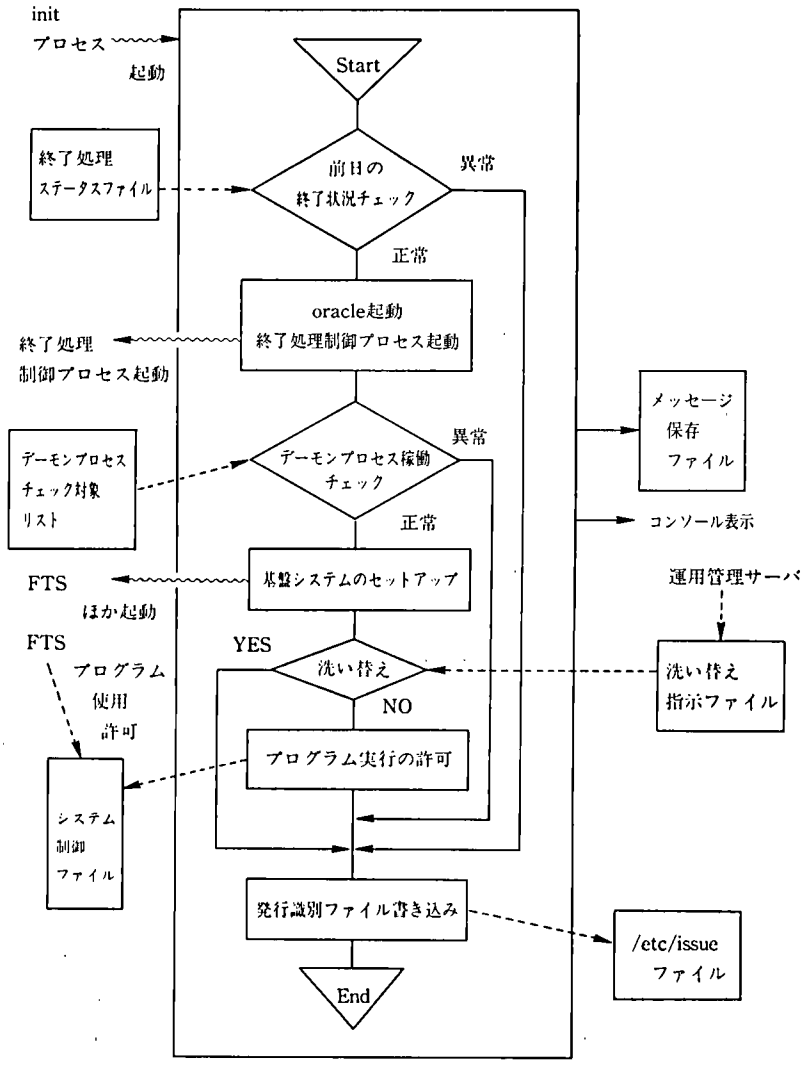


図 3　セットアップ処理フロー

は，NSS が rsh コマンドを使用するため UNIX のランレベル 2(マルチユーザ状態)に移行した時に起動するように設定している．

init プロセスによって起動されるセットアップ処理を図 3 に示す．

部門サーバの運用の自動化に伴い終了処理 (4 章参照) の結果の検査はセットアップ処理の最初のステップとして処理される．

デーモンプロセスの稼働チェックでは UNIX のプロセスやデータベースソフトウェア，コミュニケーション関係のソフトウェアの常駐プロセスが動作していることを調べる．調べる対象のプロセスはデーモンプロセス対象リストと呼ぶファイルにプロセス名を登録しておく．これにより，対象プロセスが将来増加する場合に対応できるように考慮している．

現在登録している対象プロセスは次のとおり．

pageout	メモリ制御
fsflush	ファイル情報のディスクへの書き込み
lpsched	印書制御
inetd	インターネットサービス
errdemon	システムエラーロギング
int. rwhod	ネットワーク上のログオン情報収集
in. route	ダイナミックルーティング
cron	自動プロセススケジュール
xdm	X サーバ
pcud	PCU（自動電源装置）ソフトウェア
nfsd	ネットワークファイルシステム（NFS)
biod	ブロック i/o サービス
statd	ファイルステータス管理
mountd	リモートファイルマウント
lockd	プロセスのメモリのロック，アンロックサービス（スワップ制御)
isdca	UNISYS 2200 メインフレームとの DCA 通信
istin	isdca で起動されるトランスポート層(TCP)サービス
oracle	ORACLE（データベースソフトウェア)

基盤システムのセットアップでは，FTS などのデーモンプロセスを起動し，さらに NSS を起動してオブジェクト配布を行う．運用管理サーバから洗い替えの指示があった場合は業務プログラムの使用を許可しない．この制御は，プログラム自らが実行の最初にシステム制御ファイルを参照して実行の可否を判定するようにしている．洗い替えとは，FTS を利用してメインフレームホストから受信するデータをサーバのファイルに反映させる必要がある時，そのコード体系が変わる場合（組織コードは年度ごとに変更になる）をいう．このデータをサーバのファイルに反映させるまでは業務プログラム利用を停止しなければならない．洗い替えの指示は運用管理サーバで情報システム部門が行う．また，洗い替え時には，メインフレームホストで対象データの終わりに当該データの終わりを指示するファイルを送信するよう FTS に要求し，そのファイルの受信をもってシステム制御ファイルを「プログラム使用許可」に変更して

いる．

セットアップ処理が終了する時，セットアップ処理が正常に終わったか異常があったか，を発行識別ファイル(/etc/issue ファイル)に書き込み，コンソールにも表示する．もし，UNIX のランレベル2以前でシステムが停止している場合には，この発行識別ファイルが表示されない．したがって，部門サーバの運用管理者は，コンソールに当該メッセージが表示されているか否かでセットアップが終わったか，あるいは異常があったかを知ることになる．セットアップ処理の結果は運用管理サーバに通知され，処理の経過はコンソールに都度表示されると共にメッセージ保存ファイルに書き込まれる．メッセージ保存ファイルは障害解析時の障害発生箇所の把握のための資料として情報システム部門が使用する．障害の把握方法については5章「部門サーバの稼働監視」で説明する．

4. 部門サーバの終了処理

4.1 終了処理の作業

サーバの終了時には，

- ディスク障害時の復旧作業に備えるため，各種ファイルのバックアップ
- 基盤ソフトウェアの終了処理
- 必要に応じて臨時に行うデータベースの再編成

を行う．バックアップの処理時間はソフトウェア以外の主にデータファイルをディスクと磁気テープ（DAT；Digital Audio Tape）に採るため，おおよそ1.5〜2時間を要する．毎日のファイルのバックアップ処理の自動化はセットアップ処理と同様，運用管理者の運用負荷の軽減に貢献している．

4.2 終 了 処 理

部門サーバの終了処理は以下の処理からなっている．

- プログラム使用の禁止
- ファイルのバックアップ
- データベース再編成
- メッセージの保存
- 部門サーバの電源切断

部門サーバの終了処理は，「3.2節のセットアップ処理」で起動される終了処理制御プロセス（図3参照）によって起動される（図4）．

終了処理は，まず，バックアップ対象ファイルが更新されるのを防ぐためシステム制御ファイルを変更し，ワークステーションの業務プログラムから使用されるのを禁止する．プログラムはその実行に当たって最初にこのシステム制御ファイルを参照して実行許可があるかを調べることにしている．

次に，各種ファイルをバックアップ用のディスクにコピーし，さらに同じファイルをテープにもコピーする．データベース再編成は，データベース内のフラグメンテーションや断片化を修正することで，使用可能な領域を有効利用するために行われる．データベースの再編成の実行の指示は当日の終了処理開始時刻までに運用管理者が行う．データベースの再編成実施の必要性については，定期的（年一回）に情報システ

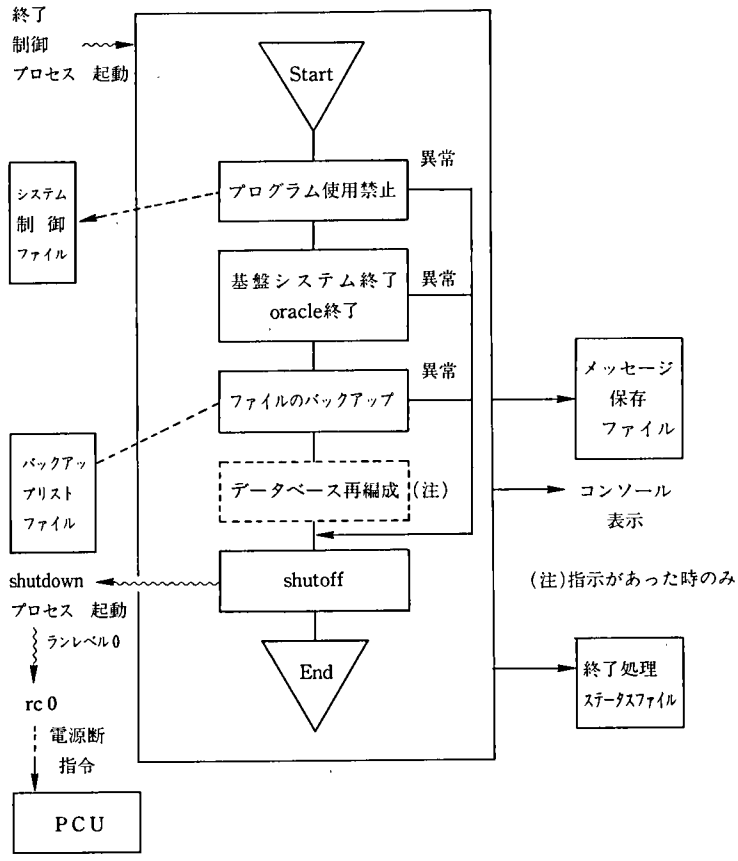


図 4 終了処理フロー

ム部門が遠隔操作にて oracle の表領域を調査し助言する.

終了処理ではセットアップと同様に途中の処理で異常が発生すると，以降の処理をスキップ（ただし，電源の切断は必ず行う）し，処理の経過はコンソールに都度表示すると共にメッセージ保存ファイルに書き込む．各処理の結果は終了ステータスファイルに記録され，翌日のセットアップで検査される．終了処理で障害が発生した場合，翌日のセットアップを抑制し，二次障害の発生を防止している．(たとえば，oracle の終了処理で異常終了しているにもかかわらず，翌日 oracle を起動してしまうと表領域を破壊しかねない.)

ファイルのバックアップは，バックアップリストファイルに登録されたファイルを対象にする．将来対象ファイルが増えた場合はこのリストに追加すればバックアップを行えるように考慮している．このバックアップリストファイルにはファイル名，キー，バックアップディレクトリ名を指定する．キーとは，リストア時に同期をとって復元しなければならない複数のファイルを一括するためのグルーピングの意味を持っている．たとえば，C-ISAM ファイルの「.dat」「.idx」「.log」ファイル等は一緒にバックアップを採り，復元しなければならない．バックアップディレクトリ名は，対象ファイルのディレクトリが複数にまたがっている場合を考慮するためである．

対象ファイルはすべてバックアップ用ディレクトリにコピーし，その後バックアップ用ディレクトリ以下すべてのファイルをDATにセーブする．コピーにはcpioを使用して，もとのファイルの属性情報（所有者/グループ/モード）とパス名を保障する．ディスクとテープとに二重にバックアップしているのは，バックアップ用ディスクの破壊とファイルの復元時間を考慮したためである．つまり，ディスクにしかバックアップがない場合，バックアップ用ディスクの破壊により，復元の手段を失うことになる．一方，テープにしかバックアップがないと，テープからの復元はディスクに比べて2〜3倍時間がかかる．バックアップは，ソフトウェアファイルについてはソフトウェアがテープで提供されるためバックアップを採らなくても復元可能であるので対象外としている．バックアップは基本的にデータファイル（oracleの表領域やC-ISAMの「.dat」ファイルなど）を対象にしている．

データベースの再編成は，oracleデータベースとC-ISAMファイルを対象としている．oracleデータベースの再編成はoracleのexportコマンドでoracleデータベースのデータをUNIXファイルに出力し，その後，exportしたUNIXファイルをoracleのimportコマンドでoracleデータベース内にロードし直している．oracleのデータベースの再編成処理には対象表領域を個々に指定する方法と，全表領域を指定する方法がある．将来，表領域の追加や削除が行り得ることを考慮し，後者の全表領域を指定する方法を採用している．C-ISAMファイルの再編成は同ソフトウェアで提供されている関数（iscluster）を使用したCプログラムで行っている．

サーバの終了処理の最終ステップは，shutoffコマンドを使用して電源を切断することである．実際にPCUに対する電源切断要求はrc 0が発行する．

5. 部門サーバの稼働監視

5.1 監視の対象

障害にはハードウェアや基本ソフトウェアに起因するものからネットワークに関するもの，あるいは業務システムに関するものまで多岐にわたるが，部門サーバの監視は，とくに次の4項目について運用管理サーバを用いて集中管理する．

- サーバの稼働監視
- セットアップ/終了処理の状況
- デーモンプロセスの稼働状況
- 業務プログラムの稼働状況

5.2 監視の方法

監視の方法は運用管理サーバから一定時間間隔で部門サーバの状況を検査するタイプと，障害が発生する都度部門サーバから運用管理サーバへ通知するタイプの二通りを用意している．便宜上，前者を能動型，後者を受動型と呼ぶと，能動型タイプは主に部門サーバのセットアップ処理状況を監視するのに用い，受動型タイプはセットアップの異常とセットアップ後の業務プログラムの異常を検知するために用いる．能動型タイプは，頻繁な監視を行うと運用管理サーバの負荷やネットワークのトラフィック増大につながるためタイムリな監視を行い難い．また，受動型タイプは部門サーバのUNIXやネットワーク関連ソフトウェアが異常な場合やネットワークの切断の場

表1 部門サーバの稼働監視方法

監視タイプ	目　　的	対　　象
能動型	UNIX システムの運行状況把握	部門サーバ稼働監視，常駐プロセス監視 セットアップ状況監視，終了処理状況監視，ネットワーク不通の検出
受動型	業務システムの異常検知	終了処理障害/セットアップ障害検出 プログラム障害(サーバ，ワークステーション)検出

合には障害を検知できない．これらの欠点を補い合うために能動型と受動型を併用している（表1）．

これらの監視状況は運用管理サーバにつながったX端末をモニター画面として利用し全部門サーバの稼働状況を表示し，異常時にビープ音を鳴らすことにより情報システム部門の監視担当者の注意を喚起させるようにしている．

(部門サーバの稼働監視は，ネットワークの監視も含めてネットワーク監視サーバにまかせているため，実際は，セッション層からアプリケーション層レベルの通信検査のみに限定している．)

伝達の経路を図5に示す．

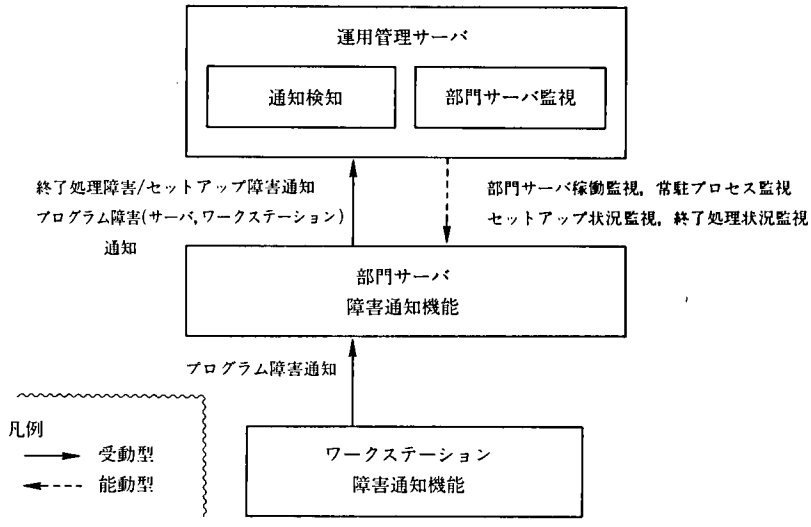


図5 部門サーバ稼働状況監視概要図

6. おわりに

ダウンサイジングが叫ばれる中にあって，運用管理の重要性の認識はいまだ乏しいといわざるを得ない．コンピュータ運用管理に従来のメインフレーム並の運用作業を適用しては，ダウンサイジングのメリットを相殺しかねないし，かと言ってまったく障害対策に無関心でもいられない．コンピュータ（部門サーバ/ワークステーション）の運用管理は，基幹業務システムの運用上避けられない．しかし，システムが提供できる運用に関わる機能は，あくまでも運用支援の範囲であり，分散設置されるサーバ

やワークステーションの正常な運用は最終的に各部門の管理体制がしっかりしているかどうかに左右される．このため運用管理に関する啓蒙と運用管理者の教育を今後とも継続する必要がある．また今後は，部門の運用管理者のための運用支援ツールを拡充させ，エンドユーザコンピューティングを運用面で支援ができるように改善を続けることが必要である．

参考文献

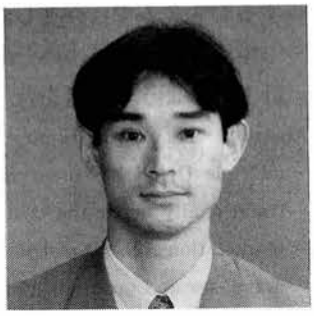
［1］ UNIX System V/386 リリース 4　システム管理者リファレンスマニュアル, UNIX システム ラボラトリーズ パシフィック（株），1991.

［2］ UNIX SYSTEM ADMINISTRATION HANDBOOK, Evi Nemeth & Garth Snyder & Scott Seebass, 1989（邦訳「システム管理入門」, 井上尚司 監訳, ソフトバンク, 1992）.

［3］ UNISYS U 6000 シリーズ SYSTEM V. 4 運用管理システム操作解説書，日本ユニシス, 1992.

執筆者紹介　内　池　貴　明（Takaaki Uchiike）

昭和 37 年生．61 年青山学院大学経済学部 経済学科卒業．同年日本ユニシス(株)入社．証券会社のシステム開発に従事．現在 情報システム部に所属．



3階層分散システムにおけるデータ配置とその維持

Data Allocation and Maintained Data Integrity for a 3-tier Distributed System

添田和宏, 宮越貞二

要 約 複数のUNIX*サーバとワークステーションを営業部門ごとに設置する新営業システムでは，従来メインフレームコンピュータで集中管理していたデータやプログラムファイルを各UNIXサーバやワークステーションに展開することになる．分散コンピューティング環境でのデータベースやプログラムファイルをどの機械に配置するか，また配置したプログラムやファイルの整合性をどのように維持するかは，メインフレームコンピュータで集中処理するシステムでは存在しなかった新たな問題である．

本稿ではファイルの分散化に伴い，ファイルの配置とデータの整合性を維持するために設計上どのような配慮を行ったかについて報告する．

Abstract The new marketing information system that obligates an individual sales department to install multiple UNIX-based servers and workstations requires the deployment of data and program files traditionally under the centralized control of a mainframe on to each of those UNIX servers and workstations. In the distributed computing environment, the determination of which units should share databases and program files and how to maintain the integrity of allocated programs and files has emerged as a new problem which has not been involved in a system where a mainframe plays a central role of data processing.

This paper describes what the authors paid special attention to, in the process of designing, to ensure proper file allocation in connection with file distribution and to maintain data integrity.

1. はじめに

新営業システムの再構築にあたり，今までのメインフレームでの一括処理のシステムにかわり，メインフレーム/サーバ/ワークステーションの3階層分散型のシステムへ変更を行った．

新営業システムは，販売事務処理を行うアプリケーションシステム（以下，業務システム），分散環境を支援する基盤システム，日々の運用管理を行う運用管理システムから構成されている．業務システムでは，基盤システムから提供される分散化された環境に対し，使用データをいかに配置していくか，業務システムとしてどのように維持していくかを新たに考えなければならない．

システムを分散化することによってシステム全体の負荷分散が得られるが，今までの一括処理と同程度のシステムの信頼性を保つためのデータの統合的な管理が必要となる．

本稿では，新営業システム構築におけるデータの分散配置方法について2章に述べ，3章で課題を整理し，4章以降で分散されたデータ維持のための業務システム設計上の

* UNIXオペレーティングシステムはUNIX System Laboratorics, Inc. が開発し，ライセンスしている．

考慮点について記述する.

2. データの分散配置

2.1 分散配置での課題

急速なハードウェアの技術革新を背景としたクライアント/サーバモデルによるシステムのダウンサイジング化により，アプリケーションとして利用する環境もより広範囲となってきている.

新営業システムにおいても，今まで一元管理されてきた各種データを分散化された環境へどのように配置し，今後運用していくかを考えなければならず，以下の点をポイントとしてデータの分散配置を行った.

- 既存システムとの接続を簡単に行える形態とする.
- 部門管理を中心として考え，今後部門固有処理を展開できるものとする.
- ネットワークへの負荷を軽減する.
- 障害時の対応を簡便に行えるものとする.

2.2 データの分類

業務システムで取扱うデータを分類すると下記の通りである.

1) 基幹データベース……販売/契約/出荷の実績をとらえるデータであり，1商内(見積，受注，契約，出荷，納入，検収)を基本単位とし，個人(営業担当者)データ/部門データ/全社データとして管理・把握される.
2) コードマスタデータベース……商品，企業等の情報に関するデータであり，業務システムでは常に参照用として利用される．他システムより1日1回更新データが提供され保守される.
3) 各種パラメタ情報……採算計算等の処理を行うための条件(為替レート，償却率等)を保持するデータであり，条件変更時のみ更新データが提供され保守される.
4) 業務システムコントロールファイル……業務システムの制御を行うためのデータである．システム日付，セキュリティフラグ，処理タイムアウト値などが含まれる.
5) 実行プログラム(ここでは，プログラムもデータとして捉える)……業務システムを実行するためのプログラムである．画面レイアウト，帳表レイアウトも含まれる.
6) ログデータ……業務プログラムの実行履歴，エラー状況を捉えるためのデータである.

2.3 データの配置方法

業務システムで取り扱うデータのメインフレームホスト，部門サーバ，およびワークステーションへの配置の考え方を以下に述べる(図1).

1) 基幹データベース

業務システムでは，部門固有の処理展開を目指しているため，基幹データベースは，部門単位に分割し管理することとした．ただし，部門データベースのみでは，以下の点において問題が発生することが予想された.

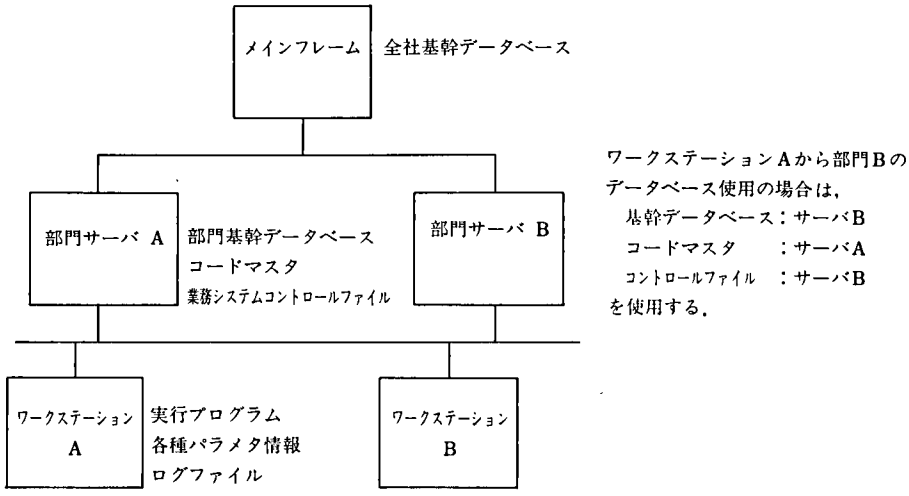


図 1　業務システムのデータ配置

① 全社管理データの即時的な把握

複数部門データベースから全社管理データを生成するのは，負荷が大きい．

② 他システム（会計部門，物流部門）へのデータ提供の集中管理

各サーバから直接他システムへデータ提供するためには，個別に状況把握が必要であり，受け側システムの対応も変更しなければならない．集中管理の面からも全社基幹データベースからの提供が妥当と思われる．

③ 障害時の対応の簡便化とデータ保証

サーバの基幹データベースに論理的な障害が発生した場合，全社基幹データベースからのリカバリが必要となる．

これらの点から，メインフレーム側には全社基幹データベースを保持し管理することにした．したがって基幹データベースの目的は，

メインフレーム：全社基幹データベースの保持
　　全社管理資料の提供，他システムへのデータ提供
　　部門サーバでの障害発生時のデータ提供
部門サーバ　：部門基幹データベースの保持
　　部門管理資料の提供

となる．

基幹データベースを部門サーバに分割配置することにより，メインフレーム/部門サーバでのデータベースの整合性の保証，部門サーバ間でのデータ移動のための機能が必要となる．

2) コードマスタデータベース

これらのデータは，主に参照用として使われるため検索速度がポイントとなる．

配置方法としては，以下の2案がある．

① コードマスタ用の一元管理サーバを設ける．

利点：運用管理が簡単である．

欠点：障害時，利用者に与える影響範囲が大きい．
ネットワーク，サーバへの負荷が大きい．

② いくつかのノードに分散し，コードマスタ用のサーバを設ける．

利点：障害時，利用者に与える影響範囲が小さい．
ネットワーク，サーバへの負荷が小さい．

欠点：複数のコードマスタの同一性を維持しなければならない．

一元管理を行うか，分散管理を行うかは相反した性格をもつが，ネットワーク負荷軽減に重点をおき，使用するワークステーションにネットワーク上，最も近いサーバ（親サーバ）へコードマスタを，ワークステーション自身に各種パラメタ情報を配置し，分散管理を行う②案とした．このため，すべてのサーバでコードマスタの同一性を維持する必要が生じた．

3) 業務システムコントロールファイル

部門サーバ配下のワークステーションに対し同時制御を行うために，部門サーバ上へ配置することとした．

4) 実行プログラム，ログファイルの配置

サーバの cpu 負荷軽減，プログラムのローディング時間の短縮，およびネットワークの負荷軽減のために，ワークステーション上へ配置した．

2.4 データの流れ

業務システムより発生するデータの流れを説明する（図2）．

1) ワークステーション→部門サーバ→メインフレーム（上り）の流れ

① ワークステーション上で稼働する業務システムは，サーバ上の基幹データベースを更新し，その更新トランザクションをメインフレームへ送信する．この際，ワークステーション上の各種パラメタ情報とサーバ上のコードマスタ情報を参照する．

② メインフレームでは，FTS（File Transfer System）から起動された業務システムのプログラムが，更新トランザクションを入力にして，基幹データベースおよび会計部門データベースを更新する．さらに業務終了後，夜間バッチ処理にて，物流関連予定情報を物流部門へ提供する．なお，FTS については，本誌別稿の「クライアント/サーバ・コンピューティングを指向した新営業基盤システム」を参照されたい．

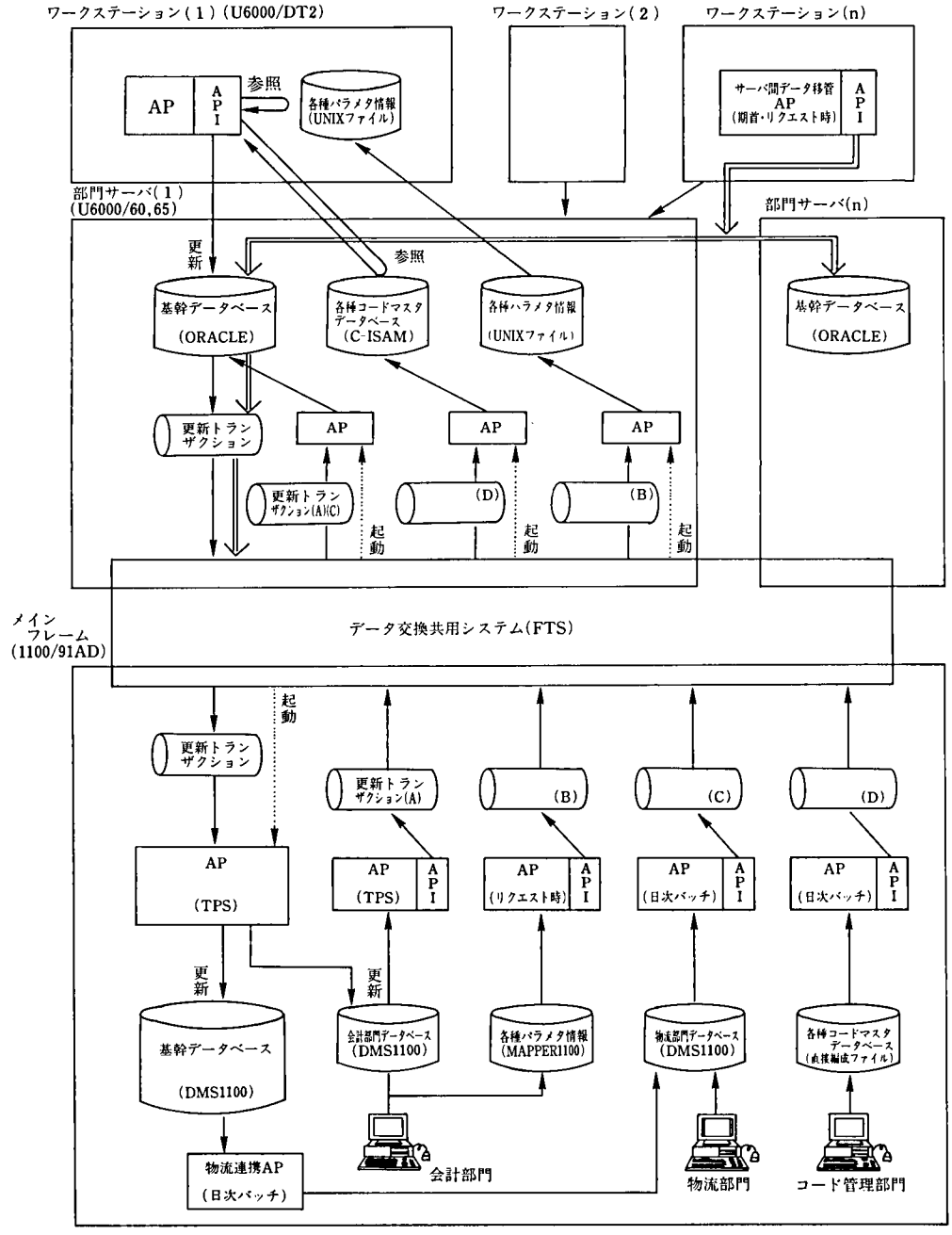
2) メインフレーム→部門サーバ→ワークステーション（下り）の流れ

① 会計部門で行う契約関連確定処理の更新トランザクションは，即時にサーバへ送信する．

② 物流部門で行う物流関連確定処理の更新トランザクションは，日次バッチ処理後，翌日サーバへ送信する．

③ その他，各種パラメタ情報はリクエスト時，コードマスタ情報は日次処理にてサーバへ送信する．

④ 各サーバに到着した①②③の更新トランザクションを入力にして，サーバ業務システムのプログラムが FTS から起動され，各データベースを更新する．



・ORACLEは米国ORACLE社の登録商標である.
・C-ISAMは米国インフォミックスソフトウェア社の登録商標である.

図 2 メインフレーン業務システム（関連システム含む）
および部門サーバ業務システム間でのデータの流れ

⑤ 各種パラメタ情報は，更新頻度が低くかつデータ量が少ないため，さらにサーバから各ワークステーションに転送する．

3) 上りの流れにおける例外処理

組織改編，個人の組織異動，その他の理由でサーバ間でのデータ移動を行う．この移動情報はメインフレームへ送信する．

3. データの分散化による課題

本章では，業務システム設計上の論理的なデータ整合性保持のための考慮点について記述する．設計上の課題を挙げると以下のようになる．

1) データベースの整合性保証

① 部門サーバ上での更新結果を正しくメインフレームに反映させる．

・転送順序の保証

-複数ワークステーションからの同一データを同時アクセスすることの防止

-代行処理の考慮

・処理を中断した時の考慮

・複数商内を同時に更新した時の転送ファイル設計

・正常処理できなかった時のリカバリ処理の自動化

② メインフレーム上の他システムとの連携を正しく行う．

・転送遅延時の考慮

③ 同一性を維持する．

・コードマスタ，プログラムの全部門サーバ/ワークステーションにおける同一内容を保証

・同一性がくずれた時の対応

④ 部門サーバ間でのデータ移動を，正しく行う．

・移動元，移動先部門サーバの容量見積

・移動処理時の整合性保証

2) データベースアクセス時の考慮

① データベースアクセスの効率化を計る．

② 複数クライアント間でのデータの共有化を計る．

・共有メモリの有効利用

③ 部門サーバ配置ファイルへのアクセスの効率化を計る．

上記1)2)につき，その考慮点を各々4章，5章で記述する．

4. データベースの整合性保証

4.1 部門サーバおよびメインフレーム基幹データベースの整合性保証

本節では，部門サーバ上の基幹データベースの更新結果を正しくメインフレームの基幹データベース上に反映させるための考慮点について記述する．

4.1.1 転送順序の保証

1) 商内単位……ワークステーション上で稼働する業務システムプログラムは，原

則，1回の処理で一つの商内単位のデータを取り扱う．商内単位とは，「顧客に対して営業活動を行う上での受注単位であり，通常1システム構成を指す.」という意味である．

2) 商内単位での転送順序保証……サーバおよびメインフレームの基幹データベースは商内単位の親情報，配下の子情報から構成されている（複数サーバ基幹データベースに同一商内単位のデータは，登録できないように考慮している）．

更新トランザクションは商内単位に作成し，受信側のメインフレームでも商内単位に処理する．したがって，更新トランザクションの一つの商内単位の中では，「サーバでの発生順序とメインフレームでの到達順序が同じであること」が要求される．

3) データ交換共用システム(FTS)……基盤システムで提供しているデータ交換共用システム（FTS）は，トランザクションを一度溜め，ファーストイン・ファーストアウトで順次処理していく待ち行列型トランザクション処理を実現している．したがって，転送指示順と到着順を一致させることができるが，これは，FTSに対してある共通キーを指定した転送ファイル群の中で有効となる．

4) 1商内単位データに対する処理制御……同一サーバ配下の複数ワークステーションから1商内単位データを同時にアクセスすることは，業務システムのプログラムでガードされている．したがって，サーバ基幹データベース上の商内単位での整合性は保たれており，更新トランザクション作成についても同様である．

5) 代行処理による転送制御……あるサーバ配下のワークステーションから他サーバの基幹データベースをアクセスすることは可能である．この際，更新トランザクションは必ずアクセスしたデータベースを保有するサーバで作られ，メインフレームに転送される．したがって，同一商内単位の更新トランザクションが異なるサーバから転送されることによってメインフレームでの到着順が保証されない，ということは発生しない．

以上の点をふまえ，今回FTSに対する共通キーとして商内単位を設定することで転送順序を保証し，メインフレームで起動されるプログラムの処理順序を保証することができた．また，異なる商内単位の更新トランザクションは，互いに独立に処理できるため，結果的にメインフレームでの処理効率改善に結び付いた．

4.1.2 処理中断時の転送制御

1) 処理中断時の仕掛制御……ワークステーション上で稼働する業務システムプログラムは，処理を途中で中断（処理放棄によるタイムアウトを含む）した場合，処理中だった商内単位を仕掛状態にする．この時，同一商内単位をアクセスしようとする他のプログラムはガードされ処理できない．

2) 更新トランザクション作成のタイミング……プログラム内で，更新トランザクション作成とFTSへの送信要求は，原則として処理完結直前で行われる．したがって，仕掛状態での転送はタイミング的に発生させないように考慮している．なお，更新トランザクション作成とFTSへの送信要求は連続して行われるため，結果的に未送信要求更新トランザクションは作成されない．

3) 取消時の例外処理……仕掛状態を解除するためには，通常，再オペレーション

をして処理を完結させる必要がある．この時，FTS への転送要求は再オペレーション処理完結直前に行われる．しかし，データ入力の誤りに気付き中止したい場合，仕掛状態を解除せずに，このデータに対して，処理未完結状態での取消処理を認めている．この場合，登録データの転送が終了していない商内に関する情報を商内単位に把握し，この情報をもとに取消トランザクションの作成を行わない考慮をしている．

4.1.3 複数商内単位更新時の整合性保証

「4.1.1 項の 1) 商内単位」で述べたとおり，業務システムでは，原則一つの商内単位データを取扱う．しかし，特定プログラムでは，二つ以上の商内単位を同時に更新する必要がある．たとえば，一つの商内単位—A の情報を部分的に，もしくは全体を他の商内単位—B に付加し，さらに付加した情報を商内単位—A から取り除く処理などである．

ここでは，二つの商内を同時更新する処理を例にして考える．

1) 転送ファイルの数

二つの商内を同時更新する場合，通常考えられるのは，各々の商内に対して一つの更新トランザクションを作成するということだが，次の問題点がある．

① FTS では一つの更新トランザクション到着時，起動されるプログラムは一つである．

② 二つの更新トランザクションの到着は，非同期である．

①については，各々の更新処理をする業務システムプログラムを分けて開発しなければならないが，決定的な問題ではない．

②については，メインフレームでの後工程処理(とくに他システムとの連携)を考えた場合，片側の更新トランザクションのみ処理したデータベースの状態は許されない．

以上の点から，部門サーバ上の 2 商内単位の更新結果は，2 商内の更新トランザクションを合成して一つの更新トランザクションにし，メインフレームに転送することにした．

2) FTS 共通キーの設定

FTS に設定する共通キーとして，二つの商内単位のどちらを選ぶか，別のキーを選ぶか検討した結果，次のようにした．

・主商内単位：主たる情報を保持する商内単位
　　共通キーに設定する

・従商内単位：主商内に従属する情報を保持する商内単位

今回の業務システムプログラムでは，以降の更新トランザクションが，従商内単位から発生する場合の処理は，複数商内単位更新の形態を採用しないで，通常の 1 商内単位の形態とすることで，それ以降の従商内としての更新トランザクション発生が起こらないような設計にしている．

3) 部門サーバでの処理順とメインフレームでの処理順の逆転

従商内単位の更新情報も含んでいる更新トランザクションが，メインフレームに到着し，主商内単位の受信キューに入る．その後，FTS から業務システムプロ

グラムが起動された時，従商内単位と同じ商内単位の前のトランザクション（受信キュー）が，未処理で残っている場合がある．本来ならば，この未処理トランザクションは，前述の従商内単位の更新情報も含んでいる更新トランザクションより，先に処理されなければならない．

4) 先行発生トランザクションに対する処理上の考慮

部門サーバ上で先に発生した従商内単位の更新トランザクションを後から処理する場合のメインフレームの業務システムプログラムの考慮点を記述する．

処理の種類によって，下記のように分類される．

① 複数商内の更新トランザクションが，先行発生トランザクションの更新内容を含んでいる場合，

すでに処理されていることになるので処理不要．

(処理順の前後を問わないケース)

② ①以外の時，

(a) 先行発生トランザクションを，主商内単位のデータとみなして処理できる場合

複数商内単位更新の一例として，従商内単位を主商内単位のデータとして全てマージし従商内単位は削除してしまう処理がある．この時，場合によっては，受信キューに残っていた従商内データを主商内データとみなして，キー置換後，処理可能なケースがある．

(b) (a)以外の時，

業務システムプログラムでは，正常に処理できない．

その状況を情報システム部門に知らしめるエラー通知機能を組み込む．

その後，業務系運用管理者による対処が必要になる．

以上の点をふまえ今回，処理順の逆転に対して業務システムプログラム内で判断できるケース（上記①，②(a)）について，処理の自動化を行った．

4.1.4 整合性保証のためのリカバリ方法

ここでは主に起動されたプログラムが何らかの理由により，正常処理ができなかった場合を想定し，エラー状況の検知とそのリカバリについて考慮した点を記述する．

1) メインフレームの業務システムプログラムでのエラー検知とリカバリ

① エラー検知の種類

(a) デッドロック等，他 TPS との競合により，処理が完結していない場合

(b) 「4.1.3 項の 4)の②(b)処理順の逆転」で正常処理不可の場合

(c) 即時連携している会計部門システムとのデータ不整合が生じた場合

(d) 転送ファイルの不具合（重複転送，サーバ側で作成ミス等）が生じた場合

が挙げられる．

② リカバリ方法

上記エラー検知 (a)～(d) に対してのリカバリは，

(a) 再実行させるため，FTS に対して再度，トランザクション発生を要求する．

これは，業務システムプログラム内で自動的に要求する．

(b) (c) 状況を情報システム部門に知らしめるエラー通知機能を組み込む．ただ

し，その後は人為的なリカバリを行う．

(d) 重複転送については対応可能な設計をしているが，他システムと連携している業務システムプログラムの中には，正常処理できないものもある．

この場合，および重複転送以外の場合は，上記 (b)(c) と同様のリカバリを行う．

2) 部門サーバの業務システムプログラムでのエラー検知とリカバリ方法

① エラー検知の種類

(a) 業務システムプログラムでのシステムエラー（主にワークステーション環境によるケース）

(b) 送・受信要求時の障害発生

が挙げられる．

現在，業務システムプログラムから呼び出される共通のエラールーチンが用意されているが，その機能は運用管理サーバに状況を通知し，データベースを復元し，エラー発生の商内単位を仕掛中にし，その後の更新を許さないようにすることである．

② リカバリ方法

運用管理サーバに通知された状況をもとに，運用管理者による対応が必要となる．

3) メインフレームからのダウンロード機能

2)の対応範囲外のケースとしての，

・部門サーバのハードウェア障害時

・メインフレームとの連携上，論理的にリカバリ不能と判断した時

について，メインフレームの基幹データベースから，部門全データ，または商内単位のデータをサーバに取り込む機能を用意している．

なお，この機能はメインフレーム，部門サーバのどちらからでも起動できる．

4.2 他システムアプリケーションの処理タイミングと転送遅延

本節では，即時に連携している会計部門でのプログラム処理が転送状況の如何にかかわらず，正しいタイミングで行われるための考慮点について記述する．

1) 会計部門での処理

部門サーバから転送された契約情報の更新トランザクションは，メインフレーム上，即時に会計部門データベースに反映させる．

会計部門担当者は，営業担当者がワークステーション上で発行した契約諸帳表を確認・照査後，メインフレーム会計部門システムにより，契約関連確定処理を即時に行う．

この結果は，メインフレーム基幹データベースに反映され，さらに部門サーバへ転送される．

2) 部門サーバからメインフレームへの転送が遅延した場合の問題点

同一商内データに対して，ある一連の契約処理を行うと，それに対する複数更新トランザクション（契約情報）が作成され，メインフレームに送信される．この時，この一連の更新トランザクションの処理を完結させた後，会計部門担当者

による処理を行わなければならない．通常状態なら，処理は完結された後なので問題はないが，転送の遅れにより部分的に処理が終了している場合，会計部門担当者にはこの状況が把握できないため，確定処理を行ってしまうケースがある(全トランザクションが未処理ならば，会計部門担当者は状況が認識できる)．

その結果，更新トランザクションが到着した時，場合によっては確定行為の取消等，人為的な処置が必要になる場合がある．

システム設計時には，このような状況は例外的であると考え，人手によるリカバリをしていたが，システム稼働当初の決算期にこの現象が多発した．

3) 会計部門での状況把握

そこで，会計部門に処理タイミングを誤らせない方策を検討した結果，次のように対処した．

ワークステーションでの契約処理進行時，発生する更新トランザクション単位に通番を採番し，その値を契約諸帳表，および更新トランザクションに書き込む．メインフレームでは，更新トランザクションの通番を会計部門のデータベースに覚え，さらに処理画面に表示させるようにした．

会計部門では，契約諸帳表と処理画面の通番一致を確認後，確定処理を行うように運用方法を変更し，徹底を図った．

その結果，現在ではこの混乱は激減しており，このような状況が発生した場合でも人手によるリカバリで対応できている．

更新トランザクションの転送状況と，通番の関係を図3に示す．

4.3 同一性の維持

4.3.1 基盤システムでの維持

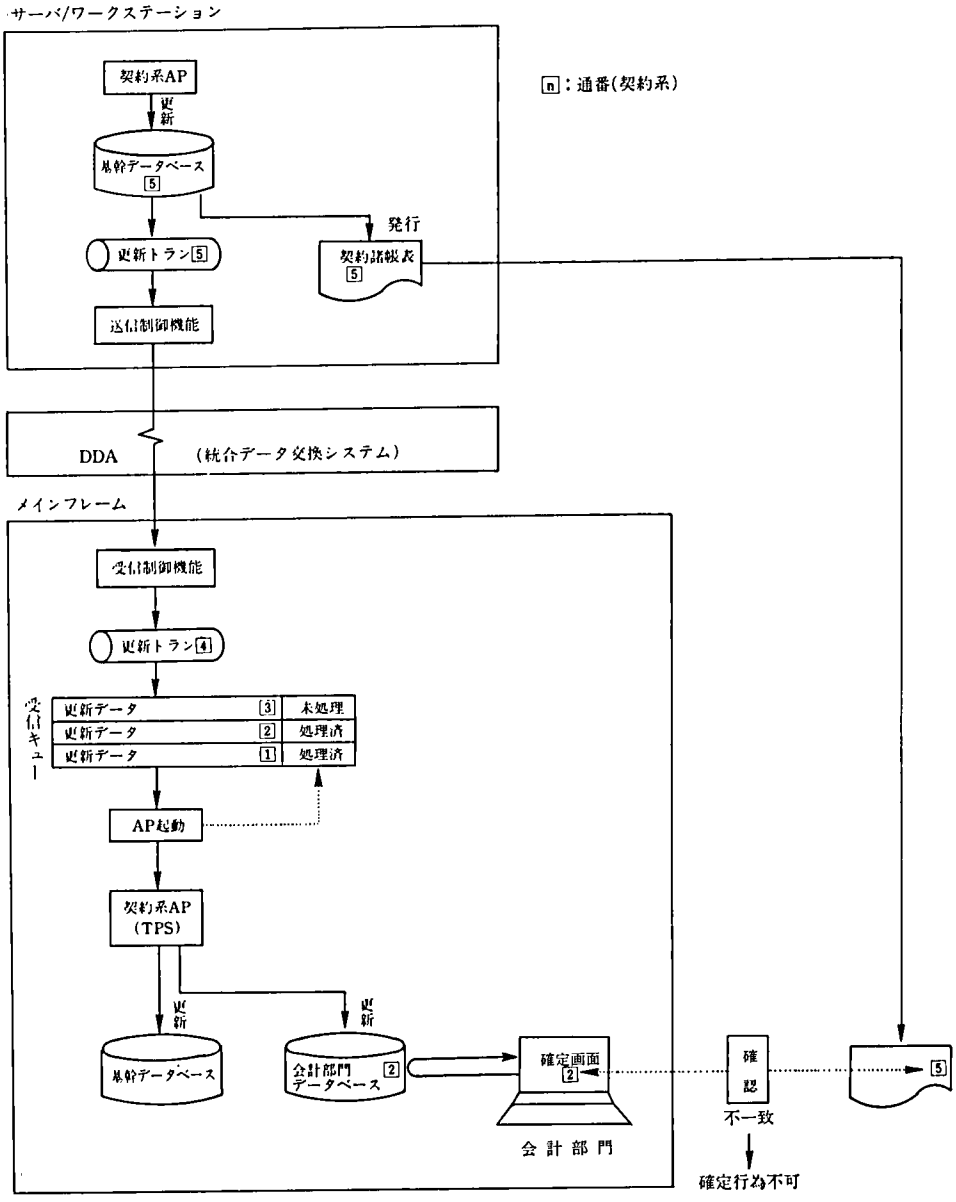
コードマスタ，各種パラメタ情報，実行プログラム等は，すべての使用者において常に同じ内容のものを提供しなければならない．

この同一性を維持するための仕組みとして，以下の機能が基盤システムより提供されている．

1) コードマスタ，各種パラメタ情報の更新……メインフレーム上で作成された更新データは，毎朝全部門サーバに送信され，FTS に自動的に登録される．各サーバでは，自動立ち上げ時に登録情報を見て各コードマスタ単位に更新プログラムを起動し，サーバ上のコードマスタを更新する．更新プログラムの不具合で処理が正常に終了しない場合は，そのコードマスタの更新のみを停止し，その他のコードマスタの更新処理は続行する．

2) 実行プログラムの更新……更新分の実行プログラムを運用管理サーバのネットワーク支援システム（NSS）に登録することにより，各部門サーバ自動立ち上げ時，運用管理サーバから更新分の実行プログラムのコピーを行う．同じくワークステーション立ち上げ時，親部門サーバから更新分の実行プログラムのコピーを行う．

なお，親部門サーバ立ち上げ時には，必ず配下のワークステーションのシャットダウンを行うため，親部門サーバより先にワークステーションが立ち上がることはない．



一連の契約行為の結果，通番＝[1]～[5]の更新トランザクションが発生したケースを想定する．

本来ならば，通番＝[5]の更新トランザクションの処理を完結させた後，会計部門担当者による処理を行わなければならない．

・通番＝[5]の更新トランザクション：サーバでの送信待ち状態
・通番＝[4]の更新トランザクション：メインフレームで受信後，受信キュー未登録状態
・通番＝[3]の更新トランザクション：受信キュー登録後，プログラムが起動されていない状態

上記，3更新トランザクションは，会計部門データベースに反映されないため通番不一致となる．

図 3　更新トランザクションの転送状況と通番の関係

また，数日間ワークステーションが立ち上がらない場合でも，未コピー分すべてがコピーされるようになっている．

これらの基盤システムの機能により，データの同一性が維持される．ただし，回線トラブル，更新プログラムのバグ，エラー検知の不具合などで同一性がくずれる場合がまれに発生する場合を想定し，対応方法を決めている．

4.3.2 業務システムでの維持

同一性がくずれた場合の業務システム側の対応について述べる．

1) コードマスタ，各種パラメタ情報……一部の部門サーバにおいて，正常にコードマスタの更新が行われなかった場合に，その部門サーバにおいてのみ更新分の商品登録ができない等の状況が発生する．

この状況を使用者に伝えるため，コードマスタ照会画面において各コードマスタの更新日付を表示することとした．

また，業務運用担当者（情報システム部門の管理者）がコードマスタの使用不可を判断した場合は，部門サーバ上の業務システムコントロールファイルのセキュリティフラグを設定することで，配下のワークステーションが誤ったコードマスタを使用することを停止させることができる．

2) 実行プログラム……一部のワークステーションにおいて，正常に実行プログラムのコピーが行われなかった場合に，ワークステーションごとに処理結果が違うという状況が発生する．

この場合，使用者側での判断はできないため，必ず業務運用担当者がワークステーションの実行プログラムの世代をチェックすることとなる．このため，世代チェック用のツール（運用管理サーバとの検証ツール）を作成し，すぐにチェックを行えるようにした．

いずれも人為的な対応であるが，今後は，全部門サーバコードマスタの更新をモニタする機能の提供，実行プログラム検証ツールの自動化を行いたい．

4.4 部門サーバ間データ移動機能

4.4.1 データ移動の必要性

部門単位にデータベースを分割したことにより，部門が変更された場合，部門サーバ間でのデータの移動が必要となる．ただし，この機能は使用頻度としては少なく，必ずしも高い処理効率は要求されない．

部門間の変更については，期末・期首時の組織改編による場合と商内の担当者が変更される場合とがあり，とくに組織改編時については，各データベースの容量をオーバしないように部門の割り振りを行わなくてはならない．

部門の割り振りについては，期末時に各部門サーバのデータベース容量を調査し，組織改編情報を新部門・サーバ対応テーブルとして営業管理部門から提供する．

4.4.2 データ移動のための考慮

1) 組織改編のためのデータ移動

各部門サーバに対し，以下の手順で作業を行う．

① 新部門・サーバ対応テーブルをもとに，移動対象商内に移動先サーバIDの設定を行う．

② 移動元の商内共通情報をもとに移動先サーバを決定し，該当する商内に関するデータをすべてコピーする．

③ コピーが成功した場合，移動元の商内共通情報にコピー済のマークを設定する．

④ 移動元のサーバより移動情報をメインフレームへ転送する．

データ移動時の効率化，移動元/移動先サーバ間での整合性，さらにメインフレームとの整合性をとるため以下の考慮を行った．

・26 サーバでのスケジュール

あらかじめ 26 サーバ間での，移動元/移動先別移動データ容量の見積りを行った．この見積りをもとになるべく同じサーバが移動元/移動先とならないようサーバを 5〜7 サーバ単位にグループ化しスケジュールした．これにより，極力サーバのレスポンスの低下を抑えた．

・サーバ間移動時のデータの保証

データ移動時（上記②③），回線等のトラブルの場合は，移動元/移動先の両データベースに対してロールバックを行う (ORACLE 2 フェーズコミット機能を使用)．

・メインフレームデータベースとの整合性

サーバ ID の変更情報をメインフレームへ転送し，メインフレーム側ではサーバ ID の変更のみを行う．

2) 担当者変更のためのデータ移動

商内担当者の変更が異なる部門サーバ間で行われた場合は，担当者情報の変更および商内データの部門サーバ間移動が必要となる．このため，担当者変更時に内部的に変更後部門サーバを割り出し，自動的にデータの移動が行われる機能を用意した（図 4)．

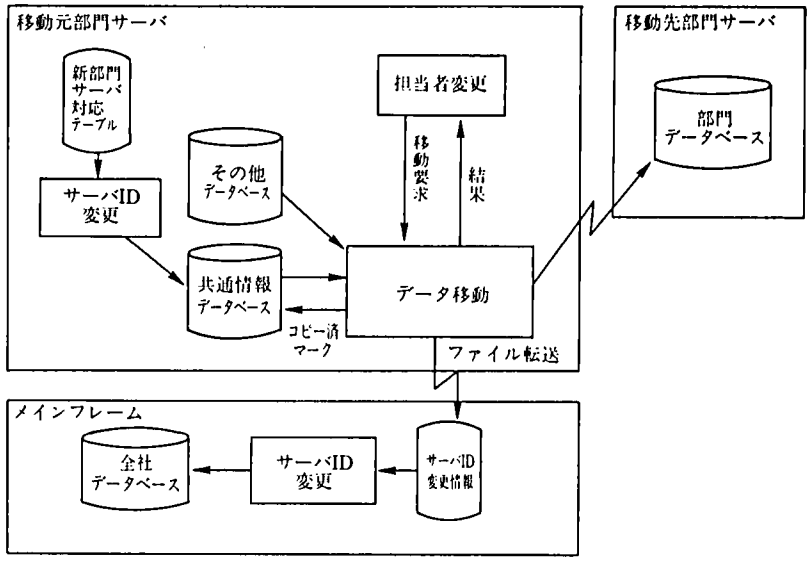


図 4　サーバ間データ移動

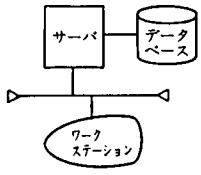
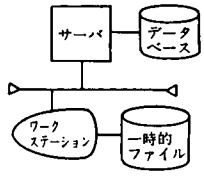
5. データアクセス時の考慮

5.1 データベース利用形態

クライアント/サーバ型システムはネットワーク結合のシステムである．そのため，今までのメインフレーム集中型のデータベースシステムとは，データベースを利用する観点も変わってくる．データベース利用に対し，発生したトランザクションをデータベース側でいかに効率よく処理するかではなく，ネットワーク上に発生するトランザクションをいかに抑え，ネットワーク負荷を軽減するかに重点が置かれる．

今回の業務システムでは，ワークステーション上の実行プログラムから部門サーバ上のデータベースをアクセスするため，データベースアクセスはすなわちネットワークトランザクションの発生となる．そのため，部門サーバ上のデータベースのアクセス方法について，表1のように二つの案を検討した．

表 1　データベース利用形態

	概念図	処理概要	障害運用	長所	短所
案1		サーバ上のデータベースを都度アクセスする．	サーバが主体	・アクセスが分散される． ・処理途中でのデータ保持ができる．	・アクセス回数が増える．
案2		サーバ上のデータベースを処理前にワークステーションへコピーし，処理後データベースへ戻す．	サーバとワークステーションで行う．	・処理自体でのアクセスはない．	・アクセスが集中する． ・処理途中でのデータ保持ができない． ・障害運用が難しい．

業務の性格上，大量データ入力時の処理途中段階のデータ保持，障害運用の簡便化より案1としたが，ネットワーク負荷の軽減の面では十分とはいえない．

このため，ワークステーション上に共有メモリを設けデータベースとの緩衝領域として利用することとした．共有メモリにより，すでに読み込まれたデータに対する再読み込みが省略され，データベースアクセスの減少になる．

ただし，アプリケーション作成者は，共有メモリを意識しながらのデータベース操作となるため，アプリケーション開発に対する負荷は大きいものとなってしまった．

5.2 共有メモリ

共有メモリは，本来複数クライアントに対してのデータの共有を目的として使用される．各共有メモリは，メモリ KEY で分割されており，業務システムは「メモリ KEY を，メモリ KEY＝ワークステーション名＋データベース区分」で設定し，1ワークステーション内の複数クライアント間でのデータ共有として利用することとした．

業務システムは，データ登録を中心とする画面（プロセス）と照会を中心とする画面，およびその二つの画面を管理制御するコントロールプロセスから構成される．二つの画面は非同期に作動するため，以下のようにデータの共有を行っている(図5)．

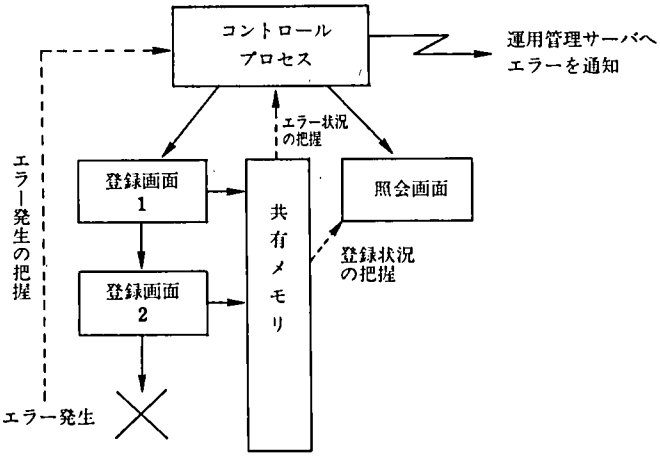


図 5　共有メモリの利用形態

① データ登録画面間での登録データの共有

② 照会画面でデータ登録状況の把握

③ コントロールプロセスでの各画面のエラー状況の把握

今後，ワークステーションへの PC 接続，クライアント数の拡大において，共有メモリの利用価値は大きいと思われる．

5.3 部門サーバ配置ファイルへのアクセス

業務システムでは，データベース，コードマスタ，各種パラメタ，転送ファイル，業務システムコントロールファイル等，多種多様のデータを扱う．特に，転送ファイルはメインフレームへの転送順序を守るために，また業務システムコントロールファイルは部門サーバ配下のワークステーションのコントロールのために，必ず部門サーバ上に配置しなければならない．そのため，部門サーバ上のファイルをワークステーション上の実行プログラムからアクセスするための手段が必要となる．

当初は，部門サーバ上のファイルアクセスのために，ワークステーション立ち上げ時に親サーバ上のファイルを NFS*により，リモートマウントして使用していた．

しかし，以下の不具合により使用形態を変更せざるをえなかった．

① データベースサーバとファイル転送サーバが同一にならない

転送ファイルを親サーバに作成すると，親サーバ以外の部門サーバのデータベースを扱う場合に，データベースサーバとファイル転送サーバとが異なる形となる．この場合，複数の部門サーバから同一商内データが転送されるため，転送順序が守られない状況が発生することがある(「4.1.1 項の 5)代行処理による転送制御」参照)．

② NFS 障害時のエラー表示ができない

業務システムプログラムでは，画面表示時に必ず業務システムコントロールファイルを参照するため，NFS で障害が発生した場合に画面が表示されない状

* NFS (Network File System；分散型ファイルシステム)：米国 Sun Microsystem 社が開発し，ライセンスしているソフトウェアである．

況が発生する．使用者には，レスポンス低下のために画面が表示されないのか障害なのか判断がつかない．

このため，新たに部門サーバ上のファイルアクセスのためのファイルインタフェース（RFCS；Remote File Control System）を作成した．

このファイルインタフェースにより，業務システムプログラムから使用サーバの切り替えを自由に行うことができる．また，ファイルアクセス時，一定時間内に部門サーバからの応答がない場合にはエラーと判断し業務システムプログラムへ制御を戻すよう考慮を加えたので，障害時に画面が表示されないという状況は無くなった．

6. おわりに

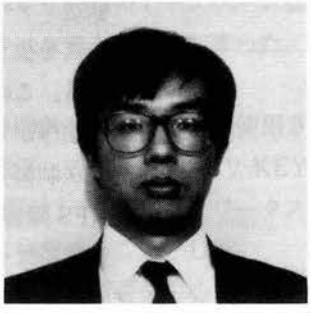
メインフレーム1台，UNIXサーバ26台，UNIXワークステーション123台からなる3階層の垂直分散型システムは，社内システムとして始めてであり，またメインフレーム/サーバ間のリアルタイム・データ連携を実現したことは，多くの課題を残しつつも評価に値すると思う．

本稿では，データの分散化におけるアプリケーション設計上の課題を提示し，解決方法を記述してきた．しかしこれはまだ一部であり，実際は業務系運用管理者が日々対応に追われているのが実情である．

今後は，EUC展開等，水平方向の分散型システムとしても発展させていかねばならないがその際，アプリケーション設計上の新たな課題に対し，積極的に取組む姿勢が必要となろう．

執筆者紹介 添田和宏（Kazuhiro Soeda）
昭和31年生．55年武蔵工業大学工学部機械工学科卒業．同年日本ユニシス(株)入社．以後社内システムの開発に従事．現在 情報システム部に所属．

宮越貞二（Teiji Miyagoshi）
昭和38年生．61年金沢大学理学部 数学科卒業．同年日本ユニシス(株)入社．客先教育に従事．現在 情報システム部に所属．



データベースのチューニングの実際

Practical of Database Tuning

鈴 木 温 雪

要 約 メインフレームコンピュータ上で構築した事務処理システムのデータベース効率を考える場合，データベースソフトウェア，オペレーティングシステム，ハードウェア領域の割当あるいはそれらの複合対応策によってパフォーマンスを向上させることができた．事務処理システムのデータベースをUNIX*上に構築する場合，ハードウェアやオペレーティングシステム，データベースソフトウェアがさまざまなメーカから提供されるため，それらの統合的なパフォーマンスを考えることは困難になっている．独立系ソフトウェアメーカのデータベースソフトウェアは，ハードウェアの追加やオペレーティングシステムの変更を当てにしないでそれ自体でパフォーマンスを改善するように作られている．したがって，パフォーマンスを含めていかにそのデータベースソフトウェアを使いこなすかがシステムの成否を握っているといっても過言ではない．

本稿は，メインフレーム上にあるデータベースを分散し，複数のUNIXサーバに多いもので数十万件のデータを格納する新営業情報システムの実例を通して，そのパフォーマンス改善の実際を報告する．

Abstract Triggered by the author's aim at raising efficiency in database performance for a business data processing system created on the mainframe, the tuning of database software and the operating system as well as the adding of hardware features, or their compound combination, has brought about success in the gaining of higher performance. The creation of a database for business data processing in the UNIX OS environment makes it hard to gauge overall system peformance bacause required hardware, operating system and database software are all supplied by different vendors. In general, database software products available from independent houses are so created as to allow 'self-enhancement' in performance without counting on any hardware additions or changes to the operating system. So, it is not too much to say that it is the key to successful system creation how well the database software can be used, including thoughts for good performance.

This paper reports on how database performance has actually been improved through the author's experience in implementing the Nihon Unisys-offered new marketing information system where a database conventionally staying on the mainframe has been distributed, and hundreds of millions of data items, to cite the largest number, are stored in a number of UNIX servers.

1. はじめに

新営業システムにおけるORACLE**データベースの設定，パフォーマンスのチューニングについて述べる．

本稿の構成は次の通りである．

2章では，新営業システムのデータベースとして必要とされる要件について述べる．

* UNIXオペレーティングシステムはUNIX System Laboratories, Inc. が開発し，ライセンスしている．

** ORACLEは米国ORACLE社の登録商標である．

3章では，新営業システムで発生したパフォーマンスの問題点を提示する．4章では，アプリケーションのデータベースアクセスのチューニングについて，目標設定，測定モデルの作成，仮説，測定，解析，対策の順に述べる．2章から4章では主にアプリケーションのチューニングについて述べるが，データベースとして必要とされる要件は，データの安全性とアプリケーションに適切なパフォーマンスを提供することである．この二点について，5章では，データベースの設定に対する要件を述べ，6章で，要件を満たすために行ったORACLEデータベースの設定について述べる．

2. 新営業システムのデータベースとして必要とされる要件

新営業システムのデータベースに対して必要とされる要件を，以下の二点に分けて述べる．

- 使用者（アプリケーション）の要求に対して速やかに応答すること
- ディスク障害やファイル破壊などデータベース障害時の回復

2.1 使用者の要求に対して速やかに応答すること

新営業システムのアプリケーションで通常使用される画面インタフェースは，約150画面あるが，この中にはcpu，メモリ，ディスクi/oなどコンピュータの資源を多く消費するもの，あまり消費しないものが混在している．また，一般に使用者が待つことのできる時間は5秒といわれている．そこで，頻繁に使用されるアプリケーションにおいて，使用者の待ち時間（ボタン押下時から次の画面が表示されるまで）を5〜10秒とすることを目標とした．

2.2 データベース障害時の回復に対する要件

新営業システムのデータベースは，全国26か所に設置されているデータベースサーバ（部門サーバ）とメインフレームホスト（1100ホスト）上のデータベースがアプリケーション上の整合性を保っている．各データベースサーバ，メインフレームホストにはそれぞれの使用者が存在し，独自に運用を行っている．新営業システムの使用者は営業マンであり，日々の営業活動の中で必要な情報をシステムへ入力している．

以前の営業システムは汎用機のオンライン処理であったので，データベースの障害回復は障害発生直前まで復旧が可能であった．

これらのことから，データベースの障害時の回復に対する要件を以下のように設定した．

- 障害の発生したデータベースサーバとメインフレームホストのデータベースはアプリケーション上の整合性を保つ必要がある．
- 一つのデータベースサーバの障害回復時に他のデータベースサーバ，メインフレームホストの使用者に影響を与えない．
- データベースを障害発生直前の時点まで回復する．

2.3 データベースサーバの制約条件

上記要件を満たすために与えられたデータベースサーバのハードウェア，ソフトウェアの環境は以下の通りである．

1) ハードウェア

U 6000/60

メモリ　32 M バイト　(ORACLE 使用可能＝約 12 M バイト)
ディスク　5 台　3.3 G バイト　(ORACLE＝840 M バイト)
(バックアップ領域＝640 M バイト)
DAT テープ装置　1 台

2) ソフトウェア

ORACLE 6.0	RDBMS	6.0.33.J 3
	TPO	6.0.33.J 3
SQL * NET	TCP/IP	1.1.J 3
PRO * C		1.3

2.4 新営業システムのデータベースについて

新営業システムのデータベースは表が 36，インデックスが 30 あり，1 表当たり 100 項目以上の表が 4，また 1 行当たりのバイト数も 400 バイト，500 バイトと大きな表が多い．項目数，バイト数の大きな表がアプリケーション上の主な表になっている．データ量は 1 データベースサーバ当たり最大のもので 80000 件となっている．

3. 新営業システムで発生したパフォーマンスの問題

新営業システムの開発作業を進めていく中で，ORACLE，アプリケーションのパフォーマンスについて問題が発生した．

新営業システムのアプリケーションのうち，最も頻繁に使用されるハードウェア商品明細表の登録・更新・削除処理に使用者の待ち時間（ボタン押下時から次の画面が表示されるまで）が 20 秒程度かかる事態が発生した．

4. アプリケーションのチューニング

4.1 目　　標

新営業システムで最も頻繁に使用されるアプリケーションはハードウェア商品明細表の登録・更新・削除処理である．このアプリケーションの使用者の待ち時間の目標を 5〜10 秒とする．ハードウェア商品明細表の登録・更新・削除処理は，①画面より入力されたデータをデータベース上のハードウェア商品明細表に反映する，②次の画面を表示する，の二つの機能に分けられる．アプリケーションプログラムにデバッグコードを入れて次の画面を表示するのに必要な時間を測定した結果，3〜4 秒が必要であることが判明した．したがって，データベースへのアクセスは 2〜6 秒で行わなければならない．

4.2 実　　測

4.2.1 実測を行うためのモデル

実測を行うために測定モデルを作成した．1 回のトランザクションで登録・更新・削除するハードウェア商品明細表の件数を知るために既存の営業システムのデータベースを調査した．1 回のトランザクションで登録・更新・削除する件数の分布は，以下のようであった．

- 30 件までの場合が全体の約 50%
- 50 件までの場合が全体の約 90%

したがって，実測モデルではハードウェア商品明細表の登録・更新・削除処理のボタン押下時から次の画面が表示されるまでのデータベースアクセスは，ハードウェア商品明細表の30件および50件の行を登録・更新・削除することとした．

4.2.2 仮　　説

実測ケースを決めるに当たって以下のような仮説をたてた．

1) データベースの母集団（すでにデータベースに蓄積されているデータ量）が効率に影響を与えているのではないか．
2) 表に設定されているインデックスの数が効率に影響を与えているのではないか．
3) 表の大きさ（1行の項目数）が，効率に影響を与えているのではないか．

4.2.3 測 定 方 法

測定は，ORACLEのトレース機能を使用して，sqlplusからsql命令を発行し，insert, update, deleteについてトレースをとった．

測定したケースは以下の通りである．

1) 母集団による差を見るために
   1500件，3000件，15000件のハードウェア商品明細表に対して各々30件，50件のinsert, update, deleteを行う．
2) 一つの表の大きさとインデックスの数による差を見るために
   母集団15000件の受託商品明細表に対して30件のinsertを測定した．

ハードウェア商品明細表と受託商品明細表の違いは以下の通りである．

- ハードウェア商品明細表
  ：150項目，600バイト，インデックスは11個
- 受託商品明細
  ：35項目，200バイト，インデックスは2個

新営業システムで設定している表を使用したので，仮説のインデックスの数と表の大きさによる差を切り分けて測定していない．

測定の前提条件として，update, deleteについてはインデックスの使用，未使用により必要な時間の差が大きいので，インデックスを使用することとし，更新する項目はインデックス項目以外とした．

実測時，LAN上の通信量が一定状態であることを確認することが難しいと考えたので，クライアントとデータベースサーバ間の通信に必要な時間は考慮しないこととした．また，ORACLEはメモリに読み込まれていないデータディクショナリが必要になるとディスクからメモリへデータディクショナリを読み込む作業を行うので，測定では，必ず，データディクショナリがメモリに存在しているようにして測定した．

4.2.4 測 定 の 結 果

測定の結果，表1のハードウェア商品明細表（30件）トレース結果，表2の受託商品明細表（30件）トレース結果の2表から次のことがいえる．

1) insertおよびインデックスを使用するupdate, delete時にはデータベースの母集団に比例して多少増加する．
2) insert時，表の大きさ，インデックスの数は効率に大きく影響を与える．

表1　ハードウェア商品明細表(30件)トレース結果

単位：1/100秒

		母集団(15000件)		母集団(3000件)		母集団(1500件)	
		cpu	elaps	cpu	elaps	cpu	elaps
INSERT	解析	349	350	360	360	344	345
	実行	191	199	169	180	178	202
	取出	0	0	0	0	0	0
	合計	540	549	529	540	522	546
UPDATE	解析	8	8	9	9	9	9
	実行	125	131	115	120	102	108
	取出	0	0	0	0	0	0
	合計	133	139	124	129	111	117
DELETE	解析	7	7	6	7	5	6
	実行	131	141	112	117	109	117
	取出	1	1	0	0	0	0
	合計	139	149	118	124	114	123

表2　受託商品明細表(30件)トレース結果

単位：1/100秒

		母集団(15000件)	
		cpu	elaps
INSERT	解析	60	60
	実行	39	46
	取出	0	0
	合計	99	106

4.3 解　　析

4.3.1 データベースの母集団の差

表1のハードウェア商品明細表（30件）トレース結果を見ると次のことがいえる．

insertのelapsの合計欄は母集団15000件，3000件，1500件のいずれの場合も5.4～5.5秒であり，1500件と15000件の差は約0.1秒である．

updateのelapsの合計欄は1.2～1.4秒で，1500件と15000件の差は0.2秒程度である．

deleteのelapsの合計欄は1.2～1.5秒となっている．1500件と15000件の差は0.3秒程度である．

insert, update, deleteとも，データベースに蓄積されているデータ量の差によって効率に大きな差はない．インデックスを使用しての表のアクセスはデータベース内のデータ量にほとんど依存しないことがわかる．ORACLEのデータベースアクセスのアルゴリズムからしても妥当な結果である．

4.3.2 表の大きさ，インデックスの数の差

表1のハードウェア商品明細表（30件）トレース結果を見ると次のことがいえる．

update, deleteに比べてinsertの解析にcpuの時間がかかっている．

母集団15000件のケースを例にとると，update, deleteのcpu解析時間が0.08秒であるのに対して，insertの解析cpuの時間が3.49秒である，解析時間に約3秒以上の差がある．

次に，表2の受託商品明細表（30件）トレース結果を見ると，insert解析cpu時間は0.6秒になっている．図1の30件insert比較図でハードウェア商品明細表と受託明細表のinsertを比較してみると，ハードウェア商品明細表のinsert解析cpu時間に比べても約2.9秒の差がある．

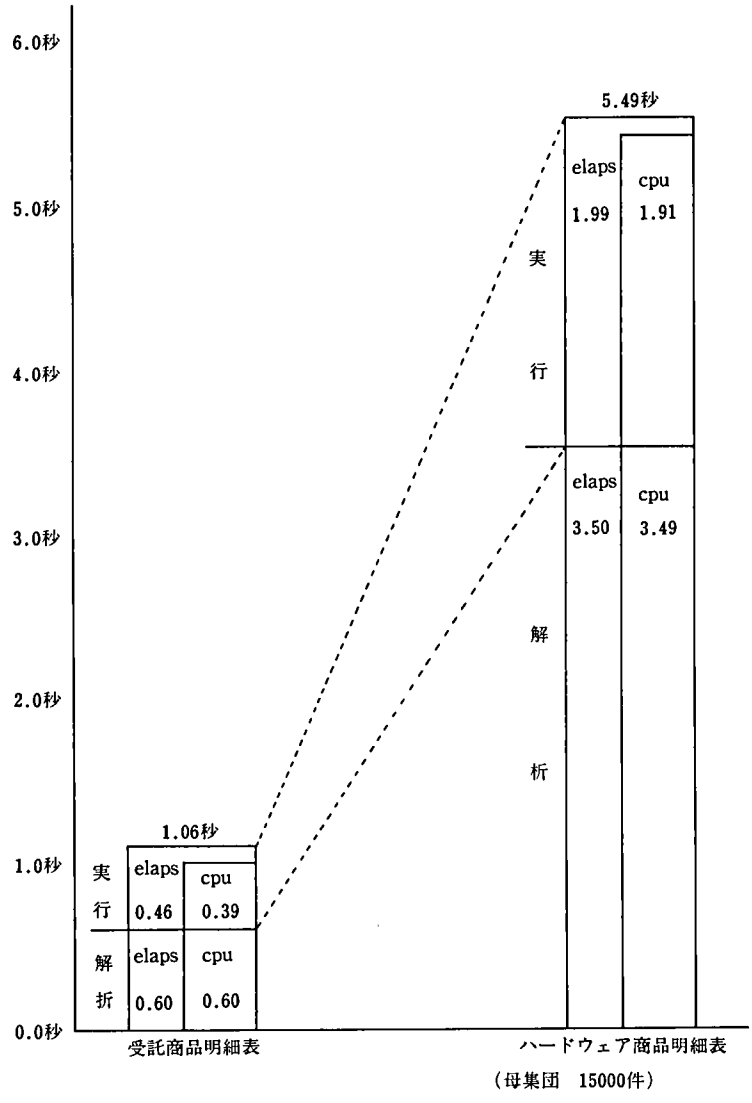


図 1 30件 insert 比較図

ORACLE の解析は，正しい SQL 文であるかのチェック，ユーザのアクセス権限のチェック，データディクショナリの検索を行い，表定義および列定義のチェックなどを行う．ハードウェア商品明細表が約 150 項目あるのに対して，受託商品明細は約 35 項目である．また，ハードウェア商品明細表のインデックスは 11 種類，受託商品明細表は 2 種類であった．項目数の差とインデックスの種類の差がとくに insert の場合に，解析 cpu 時間の差に現れていると考えられる．

4.4 仮説の検証と対策

ORACLE および ORACLE データベースをアクセスするアプリケーションのパフォーマンスのチューニングは大きく分けると，以下の四つのステップがある．

- SQL 文とアプリケーションのチューニング
- メモリ割り当てのチューニング

・入出力のチューニング

・コンテンションのチューニング

この四つのステップの詳細はORACLEのパフォーマンスチューニングガイドに記載されている．新営業システムでもこのステップに従って，チューニングを行った．ここでは，SQL文とアプリケーションのチューニングについて述べる．メモリ割当のチューニング，入出力のチューニングは6章のデータベースの設定で記述する．コンテンションのチューニングについては行っていない．

4.4.1 インデックスが多いため，insertに時間がかかる問題の対策

データベース設計時点で，インデックスの設定を行っていたが，開発が進むうちに，サブシステムやプログラムの処理の都合で，インデックスのkeyとなる項目が変更になったり，新しいインデックスが追加され，測定時点のハードウェア商品明細表では11種類ものインデックスが設定されていた．

新営業システムでは，複数の連結型インデックスをいくつかの連結型インデックスに統合し，システム全体を調査して使用している頻度の少ないインデックスは廃止して，最終的には，ハードウェア商品明細表のインデックスは4種類に絞った．

パフォーマンスチューニングガイドでは，「索引列を修正するupdate文や索引をつけた表を修正するinsert, deleteでは，索引を更新する必要があるので，索引が作成されていない場合よりも処理に時間がかかる」と記述されている．では，索引はいくつ作成すれば良いかについては，アプリケーション，データベースの設計にかかわることであるので目安などは当然記述されていない．パフォーマンスチューニングガイドの連結型索引列の選択等を参考に統合できるインデックスは可能な限り統合し，必須なものだけにするべきである．

プログラムの開発を行った後では，インデックスの統合・廃止にともない，アプリケーションの修正が必要になる．また，データベース設計後にインデックスを追加する場合は，パフォーマンスの測定を行うべきである．

4.4.2 配列処理によるパフォーマンスの向上

同一の表に対して連続して数十件の行を登録・更新する場合は，配列処理を使用することが絶対に必要である．配列処理を行うとRDBMSの1回の呼出でSQL文を何回も発行することができる．RDBMSへの呼出を減少させてパフォーマンスを向上させることができる．

配列処理の配列の数は一意には決められないが，ORACLEパフォーマンスチューニングガイドには「一般的な指針としては，配列サイズは100に設定」とあり，今回もその値が妥当であった．

sqlplusは自動的に配列処理を使用する．PRO＊Cでは配列処理の制御が用意されているが明示的にコーディングしなければならない．

プログラミングにかかわる事柄なので，アプリケーション開発時にプログラミング標準として規約を設けておく必要がある．

4.4.3 項目数が多いためinsertに時間がかかる問題の対策

対策は表を分割して，一つの表の項目数を減らすことである．

ハードウェア商品明細表（30件）のinsertのcpu解析時間は3.49秒である．受託

商品明細表 (30 件) の insert の cpu 解析時間は 0.6 秒であった．ハードウェア商品明細表の項目数は，受託商品明細表の項目数の約 4.3 倍であるので，ハードウェア商品明細表を 35 項目ごとに分割すれば，

cpu の解析時間　2.58 秒＝0.6 秒×4.3

となり，30 件の insert で約 0.9 秒縮まることになる．

これは，データベース設計時点で，よく検討しておくべきであった．パフォーマンスチューニングガイドには項目数による効率の悪化はとくに記述はないが，データベース設計時に実際に ORACLE 上に表を定義し効率を検証すべきであった．

新営業システムでは，このパフォーマンスが問題になった時点では，プログラム開発がかなり進んでおり，ハードウェア商品明細表の表の分割を実施することは，大幅なシステム変更となってしまうため表の分割はできなかった．

表 3 のハードウェア商品明細表 (50 件) トレース結果を見ると，insert 時間の合計は約 9 秒となっている．測定モデル作成時に調査したトランザクションの分布は，ハードウェア商品明細表 (50 件) の登録・更新・削除処理が 90%を占めている．インデックスの統合・廃止，配列処理の使用を行っても，ハードウェア商品明細 50 件の登録は目標を達成できない．このためアプリケーションを修正し，ハードウェア商品明細表の insert 処理を分散することにした．

ハードウェア商品明細表の登録・更新処理は，使用者が登録したデータをワークステーション上に保持し，使用者の処理がすべて終了してからデータベースの登録・更新・削除を行うようにしていた．そこで，insert のケースだけは 1 画面分 (4 行) の入力が終了した時点でデータベースへの登録を行うよう変更した．

ORACLE データベースの設計を行う場合，効率とアプリケーションプログラムの複雑さの兼ね合いになるが，一般的にはアクセス頻度の多い表の項目数は 50 項目程度

表 3　ハードウェア商品明細表(50 件)トレース結果

単位：1/100 秒

		母集団 (15000 件)		母集団 (3000 件)		母集団 (1500 件)	
		cpu	elaps	cpu	elaps	cpu	elaps
INSERT	解析	600	603	594	595	589	590
	実行	309	334	282	294	283	291
	取出	1	1	0	0	0	0
	合計	910	938	876	889	872	881
UPDATE	解析	7	7	9	8	7	7
	実行	200	207	171	184	170	181
	取出	0	0	0	0	0	0
	合計	207	214	180	192	177	188
DELETE	解析	5	5	4	4	5	5
	実行	241	250	180	193	177	206
	取出	0	0	0	0	0	0
	合計	246	255	184	197	182	211

に抑える必要がある.

4.5 対策の結果

対策の結果，現在のところデータベースサーバに接続されているワークステーションが5台程度であれば，ハードウェア商品明細表の登録・更新・削除処理の使用者の待ち時間は，5〜10秒となっている．目標は達成されているが，ネットワーク上のトラフィックや他のプロセスのcpu使用状況によっては，今後も引き続き改善する課題が出てくると考えられる．

5. ORACLEデータベースの設定に対する要件

2.2節のデータベース障害時の回復に対する要件を満たすにはデータベース障害時，以下のことを行う必要がある．

- 障害の起きたデータベースサーバにて，障害発生直前の時点まで回復する．
- 障害回復時，ORACLE管理外のファイル(メインフレームホストへ転送するファイル）と同期をとる．
- データベースサーバだけで障害発生時点までの回復ができない場合，メインフレームホストのデータベースよりデータベースサーバの回復を行う．

データベースの設定では，障害回復だけでなく与えられた環境で必要なパフォーマンスを維持することも重要な要件である．

6. ORACLEデータベースの設定

データベース設定の以下の四つのステップについて述べる．

- データベース作成
- ユーザ表領域の作成
- ユーザロールバックセグメントの作成
- メモリ割当のチューニング

なおORACLEデータベースの設定を行う場合には，参考文献[1]~[4]のマニュアルを参照する．

6.1 データベース作成

データベース作成時には，データベースの障害回復，データベースのパフォーマンスについて検討する．

データベースの障害回復については6.1.1項から6.1.7項で障害回復手順，バックアップ，REDOログファイルのモードやサイズ，ORACLEデータベースファイルのディスク配置について検討する．

パフォーマンスについては6.1.8項から6.1.9項でデータベースブロックサイズ，データベースファイルとREDOログファイルのディスク配置を検討する．

6.1.1 データベースサーバ自身での障害回復

ORACLEデータベースにおける障害には，プロセス障害，インスタンス障害，媒体障害等があるが，プロセス障害，インスタンス障害の回復についてはORAC LEが自動的に対処するので，ここでは媒体障害時対応の設定について述べる．

ORACLEデータベースは一つまたは複数個のデータベースファイル，二つ以上の

REDO ログファイル，一つまたは複数個の制御ファイルで構成されている．

各ファイルの内容は以下の通りである．

1) データベースファイル

・SYSTEM 表領域

データディクショナリや ORACLE に必要な表など

・ユーザ表領域

ユーザやアプリケーションが使用する表やインデックスなど

2) REDO ログファイル

データベースに対して行われた変更をすべて復元するためのロールフォワードデータ

・オンライン REDO ログファイル

最新のログデータを含む現在の REDO ログファイル

・オフライン REDO ログファイル

アーカイブを完了した古い REDO ログファイル

3) 制御ファイル

インスタンスがデータベースをアクセスするために必要な情報

物理的なデータベースファイル，REDO ログファイルの名称など

データベース作成作業時に，データベースファイル，制御ファイル，REDO ログファイルの媒体回復の手順を決め，次に媒体回復の手順を実現するためにはどうするか，以下の項目について検討する．

・データベースバックアップ

・REDO ログのモード

・データベースファイルのディスク配置

ORACLE の障害回復，バックアップについては，文献[1] データベース管理者ガイドのデータベースのバックアップおよび回復に記述されている．

6.1.2 ORACLE データベースファイルの媒体回復に必要な条件

データベース管理者ガイドによると，データベースファイルの媒体回復を行うための条件は以下の通りである．

・損傷を受けたデータベースファイルの障害前のバックアップコピーがあること

・REDO ログファイルをアーカイブモードで使用していること

・バックアップの時点でアーカイブを行った REDO ログファイルがあること

オンライン REDO ログファイル，制御ファイルの媒体回復を行うためには，一貫性バックアップファイルセットが必要である．一貫性バックアップファイルセットとは，データベースが停止状態（ORACLE プログラムを起動していない状態）時に，SYSTEM 表領域，ユーザ表領域，制御ファイル，REDO ログファイルを同期をとってセーブしたファイルのセットのことである．

6.1.3 新営業システムの障害回復手順

データベースファイルに障害が発生した場合は，ORACLE の通常媒体回復手順により最新状態までデータベースを回復する．

制御ファイル，REDO ログファイルに障害が発生した場合は，ORACLE は最新状態

までの回復手段を持っていないので，以下の手順でデータベースを最新状態まで回復する．

1) 一貫性バックアップファイルの時点にデータベースを回復する．
(前日業務終了時点にデータベースを回復する．)

2) メインフレームホストから1日分の更新情報を障害のあったデータベースサーバへ送付し，この更新情報をデータベースへ反映させる．

6.1.4 データベースバックアップ

新営業システムでは，以下の二つの要件を満たすために毎日のユーザ業務終了時点でクローズデータベースのバックアップを行うこととした．

・ORACLE 管理外のファイルと回復時の同期をとる．

・制御ファイル，REDO ログファイルの媒体回復に備える．

バックアップ対象ファイルは一貫性バックアップファイルセットと ORACLE 管理外の同期をとる必要のあるファイル（メインフレームホストへ送付するファイル）である．

バックアップ作業は時間指定で自動起動とし，ユーザ業務終了時にバックアップ用ディスクとバックアップ用 DAT テープの2か所へセーブしている．

6.1.5 REDO ログファイルのモード

REDO ログファイルのモードは，アーカイブモードで自動アーカイブに設定している．

回復に必要な REDO ログファイルは，REDO ログファイルの使用モードをアーカイブモードに設定しておき，オンライン REDO ログファイルが一杯になった時はオフライン REDO ログファイルへ出力するモードにしておく．アーカイブを自動的に行うのが自動アーカイブである．

設定方法は，以下のように行う．

・アーカイブモードの設定方法

データベース管理者として sqldba でデータベースに CONNECT する．

```
alter database close
alter database dismount
alter database mount exclusiv
alter database archivelog
disconect
shutdown
```

・自動アーカイブの設定

init. ora ファイルに以下の2行を追加する．

```
log_archive_dest=/～/～/arch.dbf
    (オフライン REDO ファイル用のディレクトリを指定する)
log_archive_start=true
```

6.1.6 REDO ログファイルのサイズ

新営業システムではデータベースバックアップを毎日行うことにしたので，3月末決算月（最大のトランザクション数が発生する日）1日分の REDO ログファイルが格

納できる大きさに設定した.

オンライン REDO ログファイル 1 ; 20 M バイト

オンライン REDO ログファイル 2 ; 20 M バイト

オフライン REDO ログファイル n ; 1 ファイル 20 M バイト

新営業システムのアプリケーションでは次のように前提を設けて算出した. オンライン REDO ログファイルについては, 1 トランザクションの最大の REDO ログは 3 M バイト, 同時に発生するトランザクション数は 7 として 20 M バイト. また, オフライン REDO ログファイルについては, 1 トランザクションの平均の REDO ログは 0.1 M バイト, 1 日に発生する最大のトランザクション数は 1000 とした. オフライン REDO ログファイルを格納するパーティション (REDO ログファイルのアーカイブ先) は余裕を見て 200 M バイト格納可能とした.

データベースファイルのサイズは格納するデータ量に依存するが, REDO ログファイルのサイズは, データ量ではなく, チェックポイントの頻度, 1 トランザクションの更新する量(バイト数), 同時に実行されるトランザクション数が関係する. オンライン REDO ログファイルの数は最低 2 ファイル必要である. 1 ファイルのサイズは最低 1 トランザクションが格納できるようにする. 同時に実行されるすべてのトランザクションが格納できると良い.

計算式は以下のようになる.

オンライン REDO ログのサイズ	=	1 トランザクションで作成される最大の REDO ログのサイズ	×	同時に発生するトランザクション数

オフライン REDO ログファイルの 1 ファイルごとのサイズは, オンライン REDO ログファイルのサイズと同じである. オフライン REDO ログファイルを格納するパーティションのサイズは, データベースのバックアップから次のデータベースのバックアップまでの間作成されるオフライン REDO ログファイルが格納できる量が必要である. 計算式は以下のようになる.

1 日分のオンライン REDO ログのサイズ	=	1 トランザクションで作成される REDO ログのサイズ	×	1 日のトランザクション数

6.1.7 ORACLE データベースファイルのディスク配置

新営業システムのディスク配置は以下のようになっている.

- 表領域ファイルとバックアップファイル, オンライン REDO ログファイル, 制御ファイルは別ディスクに配置する.
- 制御ファイルは ORACLE の機能を利用して 2 重化する.
- バックアップファイルはバックアップ用のディスクと DAT テープへセーブする.
- オンライン REDO ログファイルは 2 重化していない.

実際のディスク配置は以下のように三つのディスク装置に分散させている.

・表領域ファイル　　：disc 1
・バックアップファイル：disc 4
・制御ファイル　　：disc 1, disc 4
・REDO ログファイル　：disc 2

制御ファイルは多重化しておく．ORACLE は，制御ファイルの多重化機能を持っている．ORACLE では，制御ファイルが全て破壊された場合，媒体障害が発生した時点までの回復はできない．一貫性バックアップファイルセットがセーブされた時点までの回復となる．

REDO ログファイルについては，ORACLE は，多重化機能は持っていない．したがって，ミラーディスクや，ボリュームマネジャのミラー機能が必要となる．新営業システム開発時点では，U 6000/60 にミラーディスク装置，ボリュームマネージャが正式にリリースされていなかったので 2 重化は検討しなかった．もし，オンライン REDO ログファイルの 2 重化を行う場合には，パフォーマンスをよく検討する必要がある．

データベースファイルの障害については，損傷を受けたデータベースファイルのバックアップファイル，回復に必要な REDO ログファイル(バックアップ時点から，障害発生時点までの全ての REDO ログファイル)，現在の制御ファイルがあれば，ORACLE の通常回復機能を使用して，障害発生時点までの回復を行うことができる．

6.1.8 データベースブロックサイズ

データベースブロックサイズ (オラクルブロックとも呼ぶ) は，ORACLE がデータベースの入出力などを行う単位である．

新営業システムでは，データベースブロックサイズは 4096 バイトに設定した．

データベースブロックサイズは，2048 バイト，4096 バイト，8192 バイトの 3 種類があり，一つのデータベースに対して 1 種類しか選べない．また，表領域ごとにデータベースブロックサイズを変えることはできない．効率を考えると，アプリケーションが使用するすべての表の中で，1 行のサイズが最大の長さを持つ行が 1 オラクルブロックに収まるように設定する必要がある．新営業システムでは，最長の行を持つ表はハードウェア商品明細表で，約 600 バイトある．アプリケーションは連続するハードウェア商品明細表の行を扱うことが多いのでなるべく多くの行を 1 オラクルブロックに格納できるようにしたいが，30 バイト程度で行の数も少ない表もあり，大きすぎると領域の使用効率が悪くなるので 4096 バイトに設定した．

データベースブロックサイズの設定は，データベースを作成する時に決定し，データベース作成後は，データベースの再作成をしない限り変更できないので注意が必要である．

6.1.9 効率面から考えたディスク配置

ORACLE 構成ファイルのディスク配置は，媒体障害に備えた配置に加え，効率面からも検討する必要がある．基本的な考え方は，同時にアクセスするディスクを分散して効率を上げようということである．

新営業システムのディスク配置は前述のように，データベースファイル，オンライン REDO ログファイル，オフライン REDO ログファイルをそれぞれ別ディスクに配置した．

ORACLEのプロセスは常にデータベースファイルとREDOログファイルにアクセスしているので，REDOログファイルは他の動作を行わない別のディスクに配置する必要がある．また，ディスクアクセスの分散を検討する場合，ORACLEのファイル以外にも，データベースサーバ上で頻繁に稼働するプログラムファイルの配置場所なども考慮に入れる必要がある．

6.2 ユーザ表領域作成時の設定

6.2.1 ユーザ表領域の作成

新営業システムでは，以下のようにユーザ表領域を作成した．

- アプリケーションの表，インデックスを格納する表領域
- アプリケーションが使用するロールバックセグメントを格納する表領域
- アプリケーションが使用するユーザ用一時表領域，ユーザ用省略時解釈表領域

データベースの作成を行うと，SYSTEM表領域という表領域が作成される．この表領域にはデータディクショナリ，システム・テーブル，SYSTEMロールバックセグメントなどが格納される．データベース作成以降，ユーザが作成する表，インデックス，DBAが作成するロールバックセグメント，一時セグメントなどを格納するための省略時の表領域になる．

データディクショナリやORACLEが使用するロールバックセグメントなどを格納したSYSTEM表領域を通常ユーザが使用することは，効率，表領域のサイズなど不都合な面が多いので，ユーザが作成する表，インデックス，ユーザが使用するロールバックセグメント，ユーザが使用する一時セグメントは，ユーザ表領域に格納すべきである．

6.2.2 ユーザ表領域の分割

ユーザ表領域ファイルのディスク配置も，同時にアクセスするディスクを分散して効率を上げようということである．

新営業システムでは，SYSTEM表領域，ユーザ表領域，ロールバックセグメントを格納する表領域は同一のディスクに配置した．

新営業システムでは，ディスクの数は，5台であるが，データベースサーバ上では，ファイル転送システムなどさまざまなプロダクトが稼働しており，ユーザ表領域のファイルを分散配置できなかった．

できれば以下に挙げる表領域は，別ディスクに配置することが望ましい．

- SYSTEM表領域
- ユーザが使用するロールバックセグメントを含む表領域
- 頻繁にアクセスするユーザの表を格納する表領域
- 頻繁にアクセスするユーザの表のインデックスを格納する表領域
- ユーザが使用する一時セグメントを格納する表領域
- REDOログファイル

U 6000のディスクの数は通常2～5台である．媒体障害，効率の両面から検討して全てを別ディスクにすることは実際には困難であり，媒体障害からの回復を検討してディスクの配置を決め，次に効率面からのディスクの配置を検討することになる．

6.3 ユーザ用のロールバックセグメントの作成

新営業システムでは，ユーザ用ロールバックセグメントをプライベートロールバックセグメントとして，ロールバックセグメント専用の表領域に作成している．

ロールバックセグメントを効率よく使用するには，ロールバックセグメントを作成する際，エクステントのサイズが等しくなるようストレージ句を設定する．

6.4 メモリ割当のチューニング

新営業システムでは，データディクショナリキャッシュの調整，バッファキャッシュの調整を行った．データディクショナリキャッシュについては列記述キャッシュのエントリ数を2000に設定し，それ以外のデータディクショナリキャッシュはデフォルト値を使用した．バッファキャッシュについては，エントリ数を2000とした．

変更方法はinit.oraファイルのパラメータを以下のように設定した．

DC_COLUMNS=2000
DB_BLOCK_BUFFERS=2000

調整の結果，ORACLEがユーザ数に関係なく確保するメモリサイズ(システムグローバル領域）は以下のように約10Mバイトとなった..

全システムグローバル領域		9608384バイト
Fixed	Size	24504バイト
Variable	Size	736520バイト
Database	Buffers	8192000バイト
Redo	Buffers	655360バイト

適正な値を見つけるために，アプリケーションのシステムテスト中にV$ROWCACHE表，X$KCBRBH表を参照して，キャッシュミスを調査した．何度か調査した結果，キャッシュミスが多い項目についてORACLEのパラメータを変更した．

7. おわりに

新営業システムのアプリケーションのチューニングを通して，以下のようなことが言える．

- インデックスは，データベース設計時にアプリケーションでの使用を検討し，連結インデックスを使用するなど，必須なものだけに限定すること．
- 同一の表に対して数十件の行を検索・登録・更新・削除処理を行う場合は，配列処理は必ず使用する．プログラムに関わることなのでプログラミング標準などの規約を設けておくこと．
- 複数件のinsertが頻繁に行われる表については，一つの表の項目数を50程度にする必要があること．

データベースの設定では，新営業システムにおける障害回復のための設定，3種類のORACLEの構成ファイルのディスク配置，バックアップおよびメモリ割当てのチューニングなどについて述べた．

新営業システムは，ここに記述したパフォーマンスの目標は達成しているが，使用頻度，LAN上の通信量，データベースサーバで稼働している各種のプロセス等の増加により，使用者の待ち時間が長くなることが考えられる．今後も，ORACLEやアプ

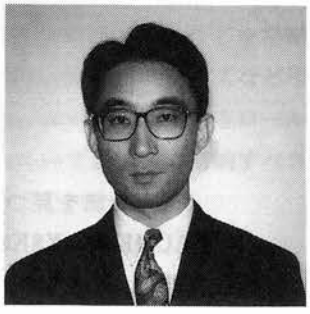
リケーションのパフォーマンスチューニングは必要であり，チューニングする項目は残っている．

ORACLE のパフォーマンスチューニングガイドに従ってチューニングを行ったが，システムに与えられた要件やハードウェア環境，アプリケーションの特性などの制限事項を前提としてパフォーマンスの要求を満たすためには，ORACLE を使用するためのノウハウが必要であり，データベース設計，アプリケーション設計時点で十分なテストを行う必要がある．

最後に，新営業システムのデータベース設定，パフォーマンスのチューニング時にご指導いただいた当社オープンシステム本部のデータベース課の方々，社会公共システム本部の方々に，感謝の意を表する．

参考文献
［1］ 日本語 ORACLE 6.0 データベース管理者ガイド．
［2］ 日本語 ORACLE 6.0 システム導入解説書．
［3］ U 6000 シリーズ SYSTEM V.4 日本語 ORACLE 6.0 システム導入解説書．
［4］ 日本語 ORACLE 6.0 ORACLE RDBMS, パフォーマンスチューニングガイド．

執筆者紹介 鈴 木 温 雪 (Noriyuki Suzuki)
昭和 33 年生．56 年日本大学商学部卒業．同年日本ユニシス(株)入社．以後社内システムの開発に従事．現在九州支店システム一部に所属．



運転チーム行動モデルの開発

The Development of an Operator Team Behavior Model

吉村誠一, 高野研一, 佐相邦英

要 約 原子力プラントのマンマシンインタフェース（制御盤）は，操作スイッチや指示計の数で1000程度もある．このような運転環境は自動化の導入，情報のCRTへの集約表示等により複雑・多様化しつつある．

このような環境下で発生する，予想外のトラブルに対する人的諸問題を検討するためには，結果としての行動を分析するのみならず，プラントにトラブルが発生してから具体的行動に至るまでの運転チームの意思形成過程を解明することが重要である．

電力中央研究所・ヒューマンファクター研究センターでは，プラントにトラブルが発生した時のヒューマンエラーの発生過程を解明し，その防止対策の立案に資するため，運転チームの意思形成過程を，個人面，チーム面から検討できる運転チーム行動モデルの開発を進めている．本稿では，モデル開発の全体計画ならびにモデルの概要について述べる．

Abstract Operational switches and indicators on the control panel (man-machine interface) of a nuclear power plant number no less than 1,000 varieties. Such an operational environment has been getting more complicated and diversified through the adoption of automation technologies, the intensive display of process information on CRTs, and the like.

The effort to study and analyze human-related problems that can occur unexpectedly in the environment of this sort requires not simply the analyzing of resultant behavior but the clarifying of the whole process ——spanning a time period from 'trouble warning' to 'specific human behavior'—— which decides an operator team to act on their intended behavior pattern.

In an effort to define the process in which human errors occur when trouble happens at a nuclear power plant so as to let the output help figure out any preventive measures, the Central Research Institute of the Electric Power Industry has been working on the development of an operator team behavior model which serves to analyze the process of formation of a team's will to act from both an individual member's and a whole group's viewpoints. Besides describing all the plan, to create the model, this paper sketches it out as well.

1. はじめに

原子力プラントは，原子炉系，タービン系，給・復水系など多くの系統があり，それらは原子炉や各種配管，弁，ポンプなど数多くの機器で構成されている．これらの機器は中央制御室の操作スイッチや指示計（これらをマンマシンインタフェース(MMI)とよぶ）で操作・監視される．原子力プラントは，このような操作スイッチや指示計の数が1000程度もある（電気出力100万Kwクラス）巨大なシステムである．運転員はこのような環境の中で日夜，原子力プラントの安全・確実な運転に努めている．しかし，些細であっても運転員のミス（ヒューマンエラー）や機器の不備・不具合が重なれば大きな事故につながる可能性がある．たとえば，1979年アメリカで発生したスリーマイルアイランド（TMI）原子力発電所の事故は，対応操作時の幾つかの

ヒューマンエラー（蒸気発生器への補助給水系の弁が閉じていたことに気づかなかったこと，原子炉水位が低下していたのに非常用炉心冷却ポンプを停止したこと，など）や計器の不備（加圧器水位計が冷却材沸騰時の水位を正しく示さなかったこと，など）により事故が拡大したといわれている[1]．

原子力プラントの運転環境は，TMI 事故後，各種計算機技術の進歩に伴い急速に変化しつつある．たとえば，運転操作は運転員の負担軽減のため，自動化される方向にあり，その一環として，異常時運転支援システムなどが導入されるようになってきた．また，指示計などの各種情報は，CRT への集約表示が進んでいる．このような変化は，運転員の操作時のミスを防ぐことを主眼にしているが，逆にシステムのブラックボックス化を招き，その結果，最新技術への適応不良が生じる可能性も指摘されている[2]．

このように複雑・多様化する環境の中で発生する，予想外のトラブルに対する人的諸問題を検討するためには，結果としての行動を分析するのみならず，プラントにトラブルが発生してから具体的行動に至るまでの運転チームの意思形成過程を解明することが重要である．そのためには，

① 個々の運転員が MMI から得られる情報をどう認知・判断し，それによりどういう行動（発話や操作）を取るかという個人面からの検討，

② 運転員間の会話や役割分担というチーム面からの検討，

が必要である．

従来のいわゆるヒューマンモデリングでは，①の個人面でいうと，外部から情報を受けた時，それに基づき自分がどう思考を組み立てるかという，メンタルモデルの考えが希薄であった．また，ほとんどのモデルは個人を扱い，②のチームという視点が欠けていた．

電力中央研究所・ヒューマンファクター研究センターでは，プラントトラブル時のヒューマンエラーの発生過程を解明し，その防止対策立案に資するため，個々人がメンタルモデルに基づき思考し，会話によりチームとしての意思を形成する運転チーム行動モデルの開発を進めている[3][4][5]．本稿では，モデル開発の全体計画ならびに次章で述べる基礎研究の中で開発しつつあるモデルの概要について簡単に紹介する．

2. モデルの開発計画

運転チーム行動モデルは，おおむね図1の流れで開発を進めている．開発は大きく基礎研究と応用研究の二つのステップに分けられる．

基礎研究は運転チーム行動モデルのプロトタイプを形成するもので，個人の認知・判断・行動を模擬するオペレータ行動モデル，個人と個人の会話などを模擬する HHI モデルならびに原子力プラントに相当する簡易プラントモデルを開発する．また，これらを組み合わせた運転チーム行動モデルプロトタイプの妥当性を実験により検証する．検証実験ならびにその結果を反映したモデルの修正は平成6年度中に終了する．

応用研究は基礎研究の成果を活用して実規模レベルの運転チーム行動モデルを開発するもので，フィールド実験により実際の運転員の知識や推論メカニズムを獲得・分析し，知識ベース化すると共に，プラントモデルとして実規模シミュレータを開発する．実規模運転チーム行動モデルの検証実験は平成8年度に行う予定であり，これに

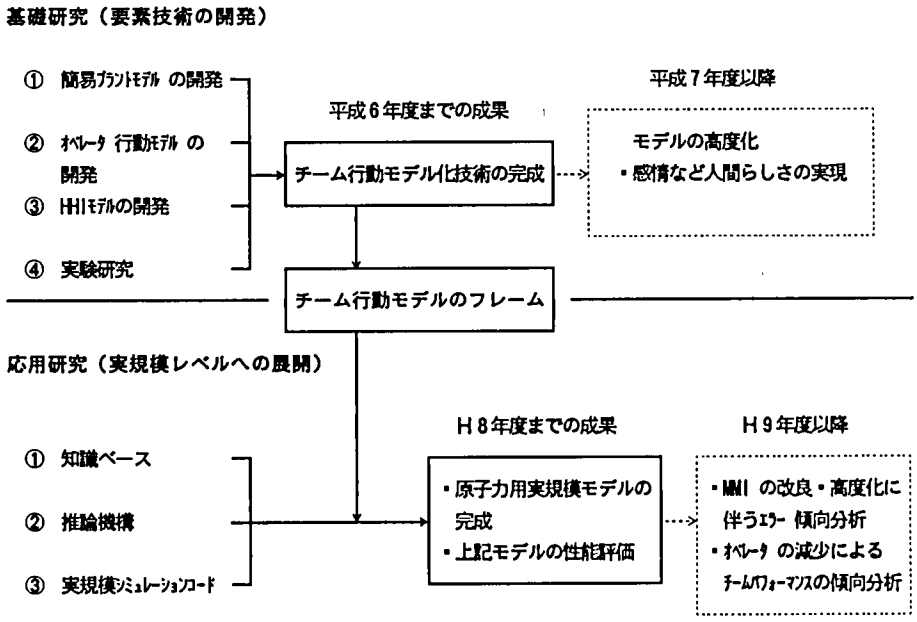


図 1　モデルの開発計画

より応用研究を終了する．

なお，基礎研究と応用研究は現在並行して進行しており，応用研究の成果（フィールド実験から得られる運転員の推論メカニズムなど）は適宜基礎研究にフィードバックする．平成 9 年度以降は，この実規模運転チーム行動モデルを用いて MMI の改良・高度化に伴うエラー傾向の分析，オペレータの減少によるチームパフォーマンスの傾向分析などを行い，プラントトラブル時のヒューマンエラー防止対策の立案に資する予定である．

3. 運動チーム行動モデル

図 2 に運転チーム行動モデルの概要を示す．運転チーム行動モデルはオペレータ行動モデルと HHI モデルで構成される．オペレータ行動モデルは一人の運転員の認知・判断・行動（発話を含む）を模擬するもので，役割分担に応じて情報（MMI 情報，発話情報）を処理し，行動する．一方，HHI モデルは運転チームのコミュニケーションを行う部分で，運転員間の意見の調整や役割分担に応じた情報の割り振りなどを決める．以下，もう少し詳しくこれらのモデルの内容を紹介する．

1)　オペレータ行動モデル……プラント，MMI を含めた運転チーム行動モデルにおける情報の流れを図 3 に示す．ここでは，オペレータ行動モデルとして 2 種類・3 人の運転員（リーダと役割分担の違う 2 人のフォロア）を考えている．

リーダは直接 MMI 情報を見に行くことも触れることもなく（動作 MM がない），警報のみが聞こえ（注意力 MM），これに基づき，フォロアに指示し（発話 MM），情報を獲得（注意力 MM）して考える（思考 MM：スキル処理とナレッジ処理に分ける）．また，フォロアは役割分担に応じて（HHI モデル）MMI 情報

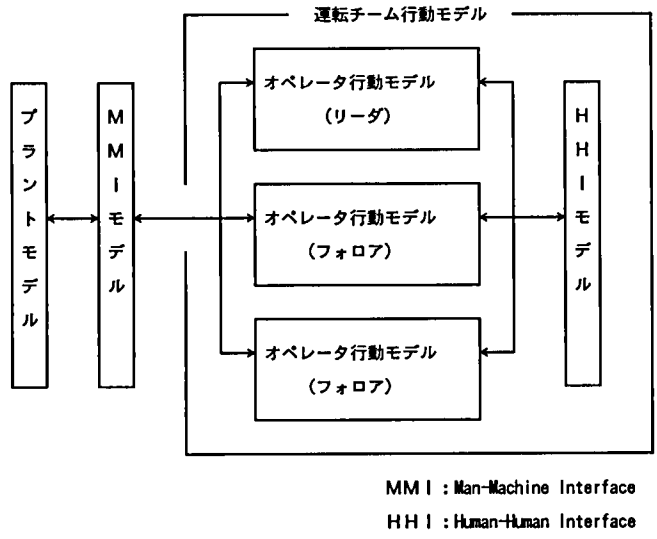


図 2　運転チーム行動モデルの概要

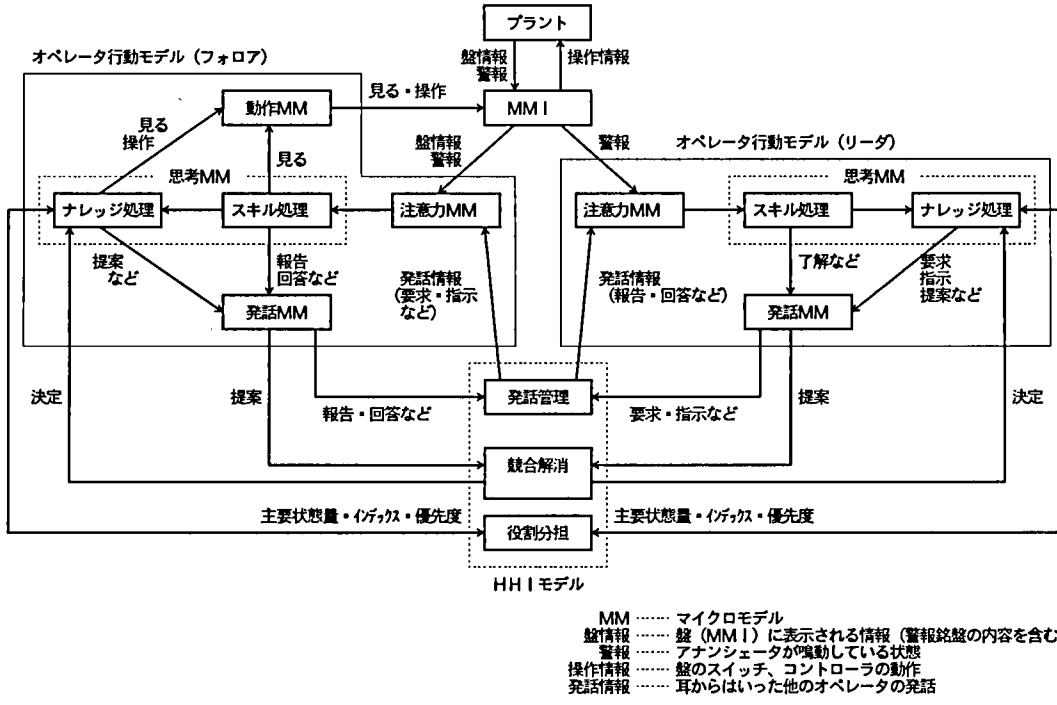


図 3　運転チーム行動モデルにおける情報の流れ

を見に行き（動作 MM），情報を獲得（注意力 MM）して考え（思考 MM：スキル処理とナレッジ処理），必要に応じてリーダと会話を行い（発話 MM），MMI を操作する（動作マイクロモデル）．このように，本モデルでは，コミュニケーションはリーダとフォロアの間だけで行う．なお，リーダとフォロア間の会話はもう一人のフォロアに聞こえるものとする．オペレータ行動モデルの各部の機能はお

おむね以下の通りである．

① 注意力マイクロモデル：注意力 MM は MMI モデルあるいは HHI モデルを経由して運転員に与えられる情報を取捨選択する機能を持つ．この取捨選択は MMI 情報の注意喚起度(目立ち易さ)，視野の種類(中心視野か周辺視野か，など)，運転員のストレスなどに依存する．

② 思考マイクロモデル：思考 MM はスキル処理を行う部分とナレッジ処理を行う部分に分ける．スキル処理は定型的な認知処理（たとえば，リーダの要求に対する回答や異常情報に対する反射的な対応（ある弁が閉まればそれを開けようとする，など）を行うもので，これを通過した情報に対してナレッジ処理が行われる．その概要を図 4 に示す．

スキル処理を通過した情報に対して，まず状況理解を行う．ここでの処理内容はおおむね以下の通り．すなわち，まず状況理解部が起動されると，中期記憶にインデックス（メンタルモデルとページで構成される）の枠が確保される．次に，このメンタルモデル部に長期記憶から関連知識がコピーされてメンタルモデルのインスタンスが生成される（メンタルモデルのインスタンスは，あることが原因となって展開した既経過あるいは今後の予測事象シーケンスの形をとる)．さらに，ページ部にスキル処理を通過した情報(あるプラント状態量に関する警報内容，操作内容ならびに値）を付加してページを生成する．なお，最も重要な監視すべき主要状態量はメンタルモデルのインスタンス生成時に，長期記憶の知識（複数の状態量がある時，どの状態量を優先して主要状態量とするかを定義したもの）を用いて決められる．このように生成したインデックスは新しい情報が得られるたびに更新される．

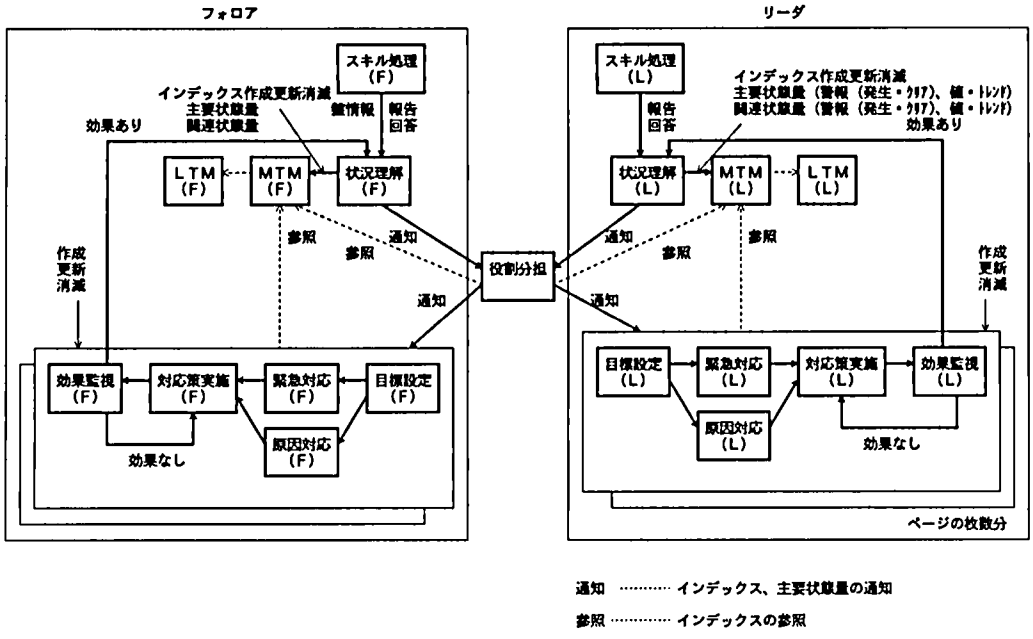


図 4　ナレッジ処理の概要

状況理解の結果はHHIモデルの役割分担部に伝えられ，その主要状態量はどちらのフォロアが対処すべきか決められる．役割分担の結果は目標設定に送られ，その主要状態量に対する対応が決められる．主要状態量が急激に変わっているようであれば緊急対応，そうでなければ原因対応を選択する．もし，リーダとフォロアの事態に対する認識状態（主要状態量としてどのパラメータを持っているかあるいはその変化を急と見るか否か，など．これらは各人の知識ベースに依存する）が違っていれば，対応策実施時に何らかのコミュニケーションが行われることになる．

③ 動作マイクロモデル：動作MMはフォロアに設けられているもので，動作意図（緊急対応あるいは原因対応）の実行や反射的な動作（警報が鳴ったら見るなど）を行う．動作意図を実行しても効果がないようであれば，中期記憶を参照しながら他の対応策を実施する．なお，リーダには動作MMがないので，フォロアが行う動作意図を注意力，思考，発話各MMならびにHHIモデルを使って監視する．

④ 発話マイクロモデル：発話MMは思考MMで形成された発話意図（対応策が間違っているのではないか，など）の実行や反射的な発話（発話を認知したことへの合図など）を行う．

2) HHIモデル……HHIモデルは図3に示すように，発話管理，競合解消，役割分担の決定などの機能を有する．役割分担は1)でも述べたように，当該主要状態量をどちらのフォロアが取り扱うべきか，また，あるフォロアがその主要状態量について緊急対応を行っているのであればもう一人は原因対応を行うなど，仕事の割り振りを行う．

一方，発話管理（声の大きさの計算），競合解消（受け入れ易さの計算）については図5のように行う．すなわち，あるオペレータ行動モデルから発話された時のHHIモデルに対する入力は，①発話内容・種類，②その発話内容に対する確信度，③受話者の発話者に対する信憑性，④発話者や受話者の覚醒度，であり，出力は発話内容・種類，受入れ易さ，ボリューム（声の大きさ）である．発話者のトーン（説得力）は発話内容に対する確信度，発話者の覚醒度，発話者の性格（権威主義度）に依存する．受入れ易さは発話者のトーン（説得力），受話者の発話者に対する信憑性，受話者の覚醒度に依存する．また，発話者のボリュームは発話者の覚醒度，性格に依存する．

各人の意見（提案）が異なる場合は，受入れ易さを指標としてどちらかが採用されることになる．

4. おわりに

運転チーム行動モデルの開発計画とモデルの概要について簡単に述べた．本モデルの特徴は，注意，思考，動作，発話など人間の認知プロセスに沿って行動する複数の運転員をモデル化している所にある．このモデルは普通の人間がそうであるように，エラーの発生メカニズムを陽な形で含まない．しかし，個人レベルでは，注意力マイクロモデルを情報に対するフィルターとして働かせることにより，各人の知識を変え

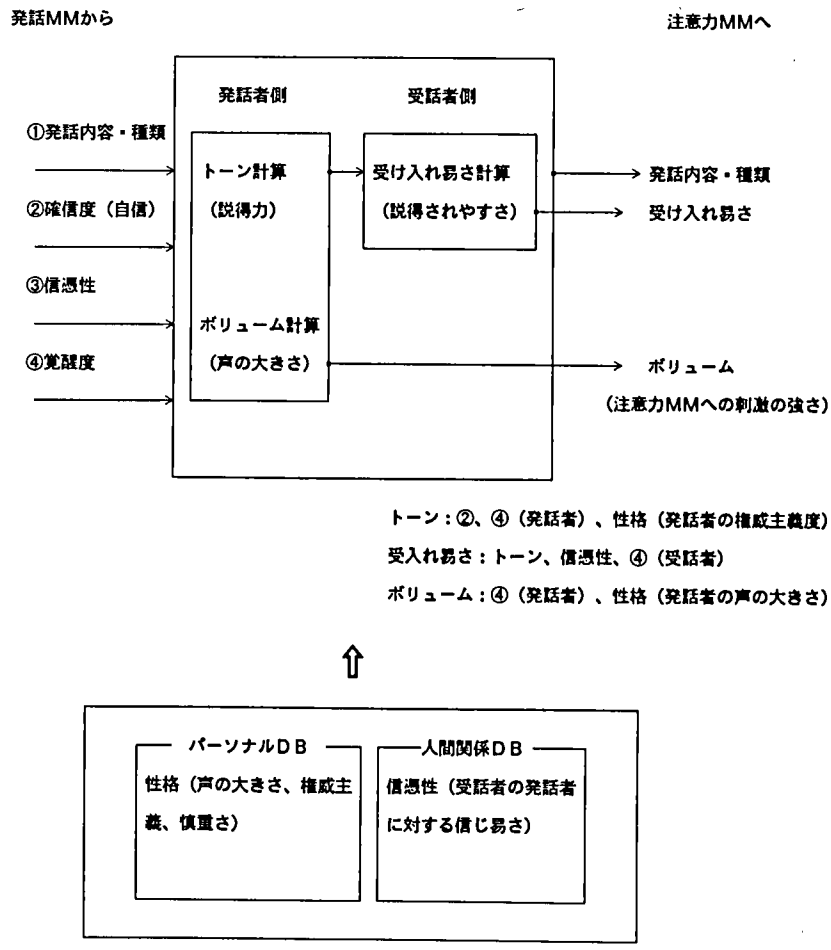


図 5　HHI モデル（発話管理，競合解消）

ることにより，また，チームレベルでは，HHI モデルに対する入力を変えることによりエラーを含む人間行動を模擬することができる．

今後，本稿で述べた考えをプログラム化し，所内実験により運転チーム行動モデルプロトタイプの妥当性を検証する．

参考文献　[1]　エネルギー総合工学研究所編集，岸田純之助監修：巨大技術の安全性，電力新報社，125-134(1987)．
[2]　稲垣敏之，誰のための自動化？，計測と制御，Vol. 32, No. 3, pp. 181～186(1993)．
[3]　吉村誠一他，運転チーム行動モデルの開発（その 1），日本原子力学会秋の大会予稿集，F 6(1993)．
[4]　高野研一他，運転チーム行動モデルの全体構成とオペレータ行動モデル，電中研研究報告，S 92001(1993)．
[5]　佐相邦英他，運転チーム行動モデル検証用プラントシミュレーションコードの開発，電中研研究報告，S 92002(1993)．

執筆者紹介　吉　村　誠　一 (Seiichi Yoshimura)

昭和26年1月1日生．51年北海道大学工学部 精密工学科大学院修士課程修了．同年(財)電力中央研究所に入所．原子力発電所保修作業の自動化・遠隔操作化，プラント診断支援システム(OECDハルデン原子炉プロジェクト)，知的教育システムの研究開発などに従事．平成4年度より，運転チーム行動モデルの研究開発を行っている．現在，ヒューマンファクター研究センター研究主幹．日本原子力学会，計測自動制御学会会員．

高　野　研　一 (Ken-ichi Takano)

昭和30年9月1日生．53年名古屋大学工学部 原子核工学科卒業．55年同大学院博士課程前期修了．同年(財)電力中央研究所に入所．保守作業用個人装備/温熱環境の生体負担評価，精神作業負荷および覚醒水準の関連生体情報評価，人的事象分析手法の開発研究に従事．現在，オペレータの思考過程シミュレーション手法の研究を行っている．日本産業衛生学会，日本人間工学会，日本原子力学会会員．

佐　相　邦　英 (Kunihide Sasou)

昭和39年9月17日生．62年慶応義塾大学理工学部 管理工学科卒業．平成元年同大学院修士課程修了．同年(財)電力中央研究所に入所．ヒューマンファクター研究センターにおいて原子力発電所の運転チームの行動を分析・評価する研究に従事．現在，運転チームのモデル化によるシミュレーション技術の研究を行っている．日本人間工学会，日本原子力学会会員．

OMTを使った「発電プラント運転員チーム行動シミュレーションシステム」の開発

The OMT-based Development of "a Behavior Simulation System for an Operator Team at a Power Plant"

羽田昭裕

要　約　電力中央研究所の委託により「発電プラント運転員チーム行動シミュレーションシステム」を開発した．操作対象であるプラントとのフィードバックを含む，チームとしての意思決定を模擬するシステムであり，先行研究・事例はほとんど無い．チーム行動は，文脈に基づく認知・判断と会話によってモデル化されると考え，この点で有効な記述方法を持つOMTを開発法として採用した．OMTの開発プロセスは，問題領域の理解，そのモデル化，設計・実装からなる．対象領域の鍵となる概念はメンタルモデルであり，モデル化のポイントはメッセージの交換と意味の整合性や文脈である．分析結果を素直に計算機表現へ転換するために，制御機構としてコルーチンと状態機械エンジンを採用した．クラスと関連の対等性，制御と機能の分離というOMTの特徴は，このシステムの分析に有効であった．離散シミュレーションとオブジェクト指向技術の親和性を示す実例となった．

Abstract　The author was on the new project to develop "a behavior simulation system for an operator team at a power plant" at the contracted request of the Central Research Institute of the Electric Power Industry. The system is meant to simulate an operator team's decision making process which includes feedback to the plant where they work, and the fact was that there were almost no preceding research papers or case studies available as reference documents. Assuming that team behavior could be modeled according to their cognition and decision based on the contexts, and spoken words the project concluded to adopt the Object Modeling Technique (OMT) method, as the development methodology, which provides the better way of representing this type of model. The OMT-based development process consists of the understanding of the problem domains, their modeling, designing and implementation. Since the key concept of the application domain is a mental model, the points for modeling include the exchange of messages, semantic integrity and contexts. To ensure that the analyzed results were straight forward transformed into computer-understandable representations, both co-routines and a state machine interpreter were used to serve as control functions. The OMT's features ——coordination of classes and their association, and separation of functions and control——have turned out to be helpful in analyzing the model. The author's experience is a good example to show similarity between discrete simulations and object-oriented technique.

1. はじめに

電力中央研究所ヒューマンファクターセンタ（以下HFC）の依頼による「発電プラント運転員チーム行動シミュレーションモデル（Simulation System for Behavior of the Operating Group: SYBORG)」の委託開発について述べる．この研究の詳細は文献[1]に述べられている．本稿は，この研究に基づくシミュレーションシステムを開発者の視点で記述する．

このシステムはダイナミックに変化する原子力プラントと，そのトラブル状態を回復する対応を行う運転員の思考過程やチームとしての意思決定過程などをシミュレーションする[1]．シミュレーションの型は，人間系の事象－事象時間による離散型とプラント系の連続型の混合型である．そして，プラントと複数の人間が同時にかかわり合う現象と個々の人間の複雑な知識構造・メンタルモデルの表現が重要である．このようなシステムを分析するには，客観的に議論できる表現方法を用い，その記述を具体例に当てはめトレースする必要がある．また，記述結果はできるだけ素直にプログラムへ写像でき，また保守性に富むことが望ましい．そこでOMT(Object Modeling Technique)[2]を選択し，開発を進めた．

2章でOMTは適用領域の本質的な部分をモデル化することを示し，3章で個人・チーム・プラントのシミュレーションシステムを構築する際の要点とその中心概念であるメンタルモデルについて述べる．4章で文脈に基づく意思決定と発話を記述する上で，OMTの動的モデルと関連の強調が有効であることを述べ，5章で分析結果を示す．6章で分析結果を素直に計算上で表現するために，コルーチンと状態機械エンジンを利用したことを述べる．その背景としてOMT，シミュレーション，有限状態機械，C＋＋の関連性に触れる．

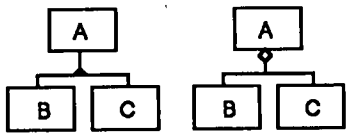
2. OMTとモデル化

OMTはその名の示すとおり，モデル化技法である．モデル化技法は，モデルの記述方法とモデルから計算機表現への変換方法を含む．

OMTにおけるモデル化の対象は，実装や機能といった偶然的・一時的なものではなく，適用領域(application domain)にとって本質的な概念である．モデル記述は，適用領域の中から識別された"もの"の属性と振る舞いを結合したオブジェクトに基づく．モデルを表現に変換するプロセスでは，まず適用領域でのオブジェクトの識別に力を傾注し，次にそれらに適合した手続きを割り当てるという手順を踏む[2]．

まずモデルを構成する概念と記法を紹介する．OMTのモデルは，オブジェクトモデル，動的モデル，機能モデルという3種類のモデルで記述する．オブジェクトモデルはクラスおよびクラス間の関連(association)を静的に表現したものである．クラスは，オブジェクトを構造の共通性によってグループ化したものである．クラスに対応する振る舞いのうち，オブジェクト間の相互関係を動的モデルで，オブジェクトが行う変換を機能モデルで表現する．3モデルの基本をなすオブジェクト図（図1）・状態図（図2）・データフロー図は，グラフ構造を図で表現したものである．

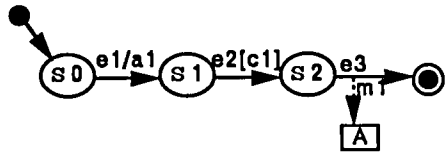
次に開発プロセスにおけるモデルの位置付けを見る．OMTでは初めに適用領域から本質的な部分を抽象した分析モデルを作成することにより，構築に先だって対象を理解する．実装の直前までプログラム言語とは独立した概念中心のプロセスを踏むことにより，設計者・開発者・適用領域の専門家などが抽象概念を明確に記述でき，またその記述に基づく討論を可能にする．この分析モデルは設計・実装に引き継がれる．設計では，システムアーキテクチャを定め，オブジェクトモデルを指針として詳細化し，設計モデルとする．最後に設計モデルを実装する．



汎化/継承図係
BとCは
Aの性質を持つ.

集約図係
BとCは
Aの一部である.

図 1　オブジェクト図の例



□初期状態はS0である.
□状態s0で事象e1が発生すると
　動作a1を実行し状態s1へ遷移する.
□状態s1で事象e2が発生したとき
　条件c1が成立すれば状態s2へ遷移する.
□状態s2で事象e3が発生すると
　クラスAに事象m1を送信し,
　終了状態へ遷移する.

図 2　状態図の例

3. 適 用 領 域

オブジェクトモデルの内容は問題に応じて決まるので，オブジェクトモデルの構成は，解決すべき問題自体の理解から始める．そこでまず，HFC での研究成果とそこから得られるモデル化の前提を述べる．

3.1 モデル化の前提

このシステムの要件は組織構造をもつ集団を，制御対象・環境を含めてモデル化し，それをシミュレートすることである．このモデルは，ミクロからマクロへ視点を移すことにより，個人の内部，個人とプラント，そしてチームとプラントという問題に分解でき，それぞれについて以下のような知見が得られている[3]．

① 個人の内部は，認知・判断と個人内差を表現する必要がある．認知・判断では，状況による意識の切り換えや予測を含めた意思決定をモデル化する．これを精緻にするには時間の概念を加味しなければならない．個人内差は，ストレスによる知覚狭窄や状況による視覚/音声情報の知覚効率の変化と，それによる見落としや聞き逃しを説明する．

② プラントは，沸騰水型原子力プラントを簡略化したものであり，原子炉・タービン・復水系・給水系から構成される．沸騰水型のプラントは，原子炉での核反応によって得られた熱で水を沸騰させ，その蒸気でタービンを回し発電する．蒸気は復水器で水に戻され，再び原子炉に給水される (図 3)．また，プラントは異常事象の根本原因であるマルファンクション(mulfunction)を発生させ，それによる一連の変化を再現できる．個人は訓練された運転員であり，中央制御室から制御盤を通じてプラントを制御する．制御盤は警報(annunciator)や状態量 (流量，圧力，水位)，機器の作動/不作動や弁の開閉といったプラント状態を表示し，機器を操作できる．

③ チーム行動のモデル化で必要なことは，リーダシップの型と個人差の表現である．個人差は，能力・慎重さ・ストレス耐性などの性格と，経験・声の大きさなどで表す．これに相手への信頼感と権威主義度を加味してリーダシップの型が決まる．これらに基づいて，複数の目標から一つを選択する，役割分担を

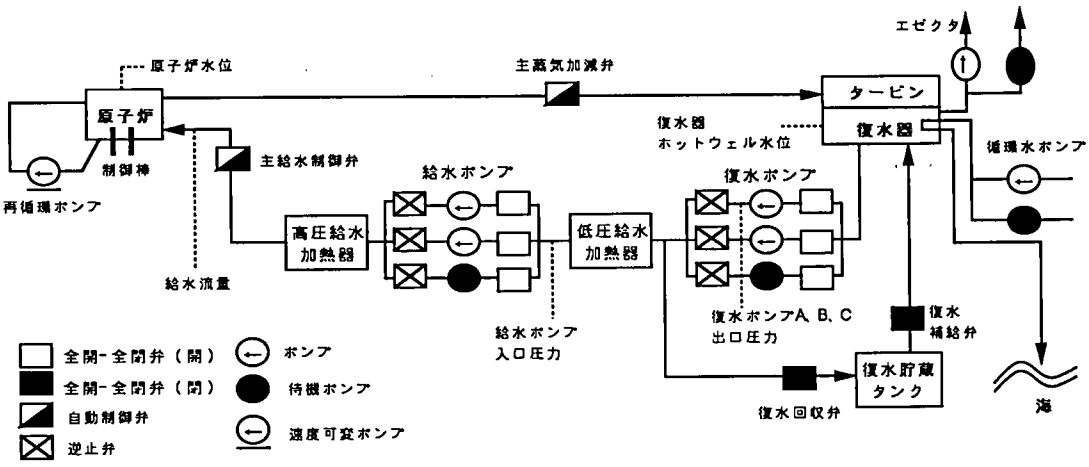


図 3　プラント系統図と状態量

するといったチーム行動が行われる．チーム行動は，個人・チーム・プラントの特性や状態によって総合的に行われる．

以上のような問題を，計算機上で実行・検証可能なモデルとするために，以下の方針を設定した．

個人の認知・判断に関しては知識処理技術を導入し，とくに意識の切り換えを表現するために次節で述べる「インデックスで整理されたメンタルモデル」という概念を導入する．個人内差は，人間の注意力を模擬することにより現実に近づける．また，プラントへの働きかけである動作と，他の個人への意志伝達である発話は，その意図形成と実行を分離することにより，思考の範囲および個人の内と外を明確にした．知識を蓄える長期記憶(Long Term Memory：LTM)と短時間で減衰し容量制約のある短期記憶(Short Term Memory：STM)の他に，推論や予測といった思考の材料や結果を保存する作動記憶である中期記憶(Middle Term Memory：MTM)を導入した．なお，この研究は運転チーム側のシミュレーション技術確立を主目的にしているため，プラントの構造は大幅に簡略化し，また進展が緩やかな異常事象のトラブル発生から原子炉スクラムまでを模擬する[4]．また，不具合によって運転系が停止すれば，待機系が稼働するといった安全のためのインターロック機構は必要最小限に抑え，運転チームの認知・判断や操作に依存したプラントとなっている．

3.2　メンタルモデル

メンタルモデルとそれに基づく意思決定についての理解のため，筆者らが参考にしたメンタルモデルと梯子モデルについて簡単に紹介し，HFC で得られたインデックスの概念を示す．

3.2.1　メンタルモデルの用語

メンタルモデルは一言で言うと心の目で対象となる機械を模擬することである．メンタルモデルは四つの基本概念から構成されると考える．四つの基本概念とは，装置のトポロジ，想見(envisioning)，因果モデル，その運用である(図 4)．装置のトポロジとは，所与の装置の構造である．想見はその装置の構造に対する機能を決定する推

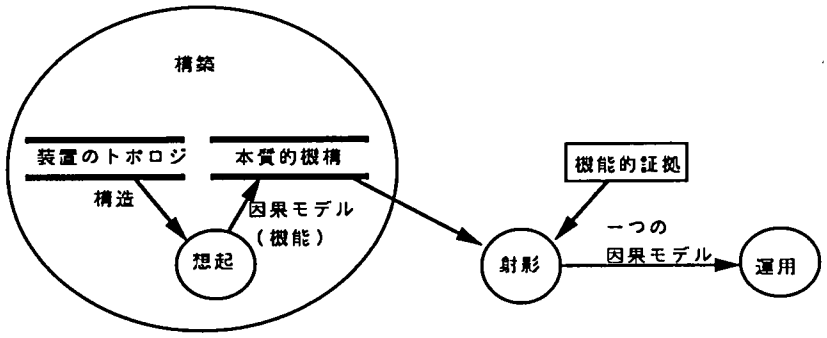


図 4　メンタルモデルの概要

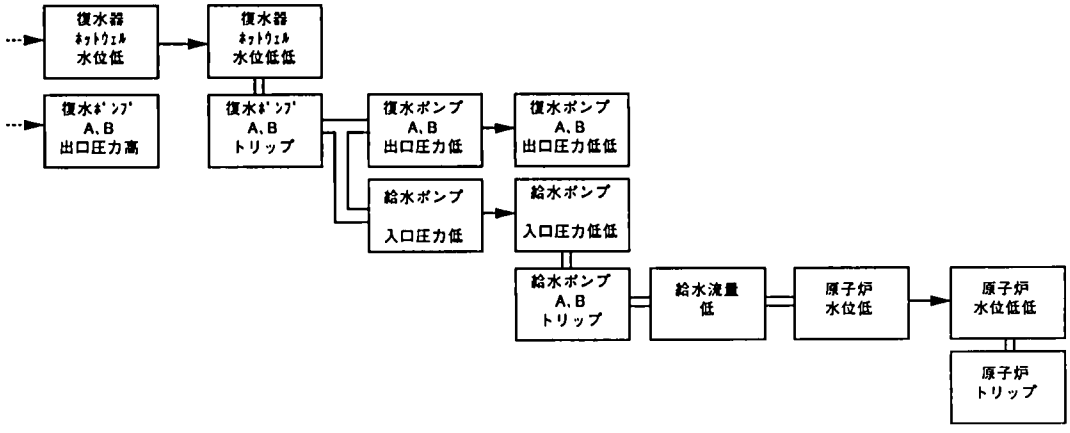


図 5　因果モデルの例

論過程であり，結論としての装置の働きが因果モデルである．こうして構築された機能を実際の状況の中で模擬するのが運用である[5]．

今回のシステムでは，各運転員は装置のトポロジとして図3のようなプラント系統を描いている．矢印の方向に水が流れており，すでに述べたようにタービンを回した蒸気は，復水器で冷却され水になる．この水は，復水器下部のホットウェルに溜まる．ここで復水器ホットウェル水位低警報が発生すると，これが機能的証拠となり，一つの因果モデルが運用される．この因果モデルは復水回収弁が誤動作により開いたというマルファンクションから想起された次のような異常事象の連鎖である (図5)．復水回収弁が開くと，復水ポンプ出口圧力が上昇し，復水器ホットウェルの水位が下がる．さらにホットウェルの水位が低低になるとインターロックにより復水ポンプがトリップする．復水ポンプがトリップすると，復水ポンプの出口圧力と給水ポンプの入口圧力が低下する．給水ポンプの入口圧力が低低になると，インターロックにより給水ポンプがトリップする．給水ポンプがトリップすると給水流量が減少し，原子炉の水位が下がる．原子炉の水位が低低となると，原子炉がトリップする．

このシステムでは，装置のトポロジ・想見・因果モデルは，長期記憶の範囲であり，そこから因果モデルを取り出し，現実の数値などを当てはめて具体的に運用することが中期記憶の働きと考えた．

3.2.2 梯子モデル

Rasumussen は，発電所の制御室で行われる意思決定の分析から，梯子モデル(ladder model)という認知的タスク分析の枠組みを提案した[6] (図 6)．対処が必要な事柄が認知されると，情報とデータを観察し，現在の状態を同定し，起こり得る結果を推測する．そして目標を選択し，それを実現するためのタスクを決め，手順を計画し実行に移す．

これらの処理は階層をなしており，またいくつかの処理を飛び越して短絡している．Rasumussen は短絡の具合により，認知・判断を三つのレベルに分割している[12]．なじみの薄い状況では，この梯子を一段づつ達成基準の評価まで昇り，実行まで下りていき，試行錯誤を繰り返しながら判断していく．これをナレッジベースの処理と呼んでいる．過去の経験などがあると，短絡した道を通り判断を加速する．これはルールベースの処理である．スキルベースの処理は，訓練などにより反射的になされる．図 6 で活性化から直接実行に移る道がそれである．

階層ごとに異なる時間尺度で機能する．高位の認知・判断は，より長い尺度で，より大きな範囲の状況に対応し，主に目標達成のための手段の提供や目標への進行度合いの監視に関連する．なお，短期記憶の制約により，一時点では一つの目標に注目すると考える．

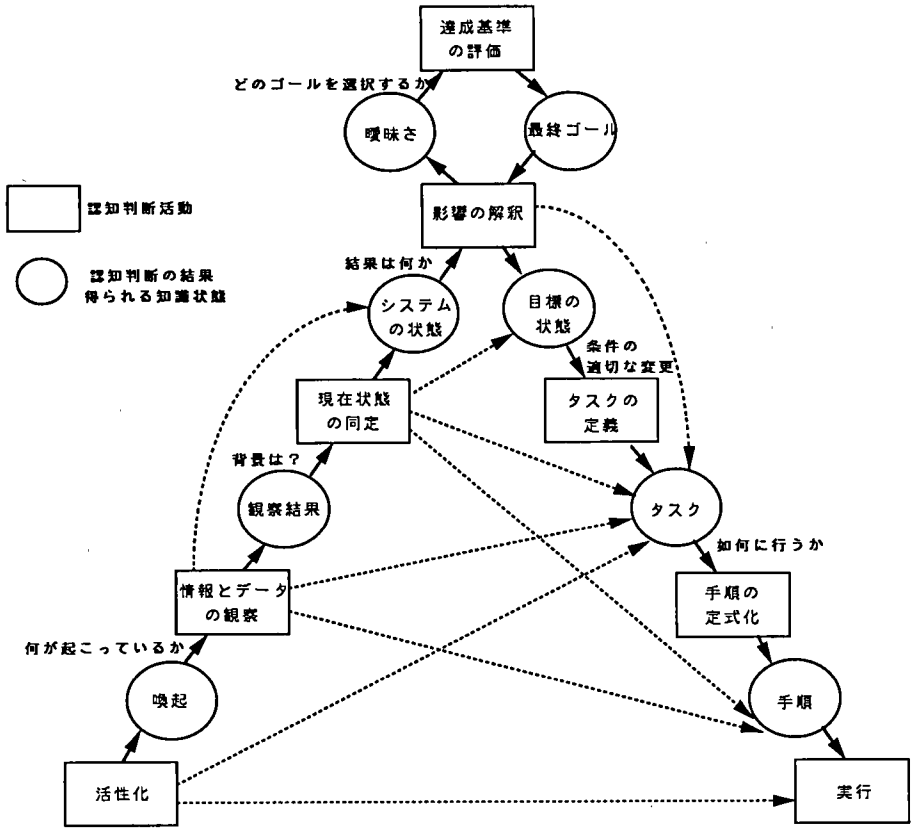


図 6 梯子モデル[6]

3.2.3 メンタルモデルの整理の仕方

複雑に見える運転員の行動は，注目している状態量によって整理でき，運転員がある時々に注意している状態量は最大二つ程度であり，多くの場合は単一の状態量の動向にのみ注意を払っているという結果が得られている．

多数の状態量の中から注目する単一の状態量を選択する判断の背景には，状態量の変化に伴う別の状態量の変化と異常事象の進展の予測，そして異常事象の重大性に対する判断があると考える．そこで一つのマルファンクションから発生する異常事象，それに伴う状態量の変化と異常事象の進展を含む予測シーケンスを一つのインデックスとしてまとめ，そこに含まれる各状態量について変化の履歴をページに記録する．現在注目している状態量を主要状態量と呼び，それに関連して動く状態量を関連状態量と呼ぶ[1][7]．

発生した警報を打ち消す対応策を当面の対応，警報設定点およびその中間のマイルストーンに用意した対応策の中で緊急度の高いものを緊急対応，同定された原因をつぶすための対応策を原因対応と呼ぶ．

インデックスは個々の因果モデルに相当すると考えた．

4. モデルの特徴と OMT

モデル化の基本は，抽象化すなわち直接取り扱うことが難しい複雑な事象を理解しやすい形にすることである．先述したように，人間の思考は一度に限られた情報しか処理できない．そこで，同時に扱う重要な事柄の数を分割して減らすことにより複雑さを減少させる．OMT はこれを三つのモデルで十分としている[2]．その中でオブジェクトモデルにおける関連の強調，動的モデルにおける階層性をもった状態図，事象フロー図，事象トレース図などの道具建てが OMT を特徴づける．この二つを今回のモデル化への有効性と共に述べる．

4.1 動的モデルの特長

先述したように，動的モデルはオブジェクト間の相互関係を記述する．これは，メッセージの送受を表現することであり，そのための図が事象フロー図である．また，これを具体的なシナリオに当てはめ，時間的前後関係を含めて記述したものが事象トレース図となる．

この観点からするとオブジェクトは，一つのメッセージを入力されるといくつかのメッセージを出力する仮想的な機械と見なせる．これは有限状態機械 (finite state machine；以下 FSM) であり，状態遷移図(state transition diagram；以下 STD) で描ける．従来の STD は，状態や遷移の数の組み合わせ爆発を防げず，並列性やモジュール性を表現できない．これらの問題を解決するために，遷移を共有する状態をスーパー状態として纏めあげるなど，STD に深さや直交性といった概念を導入したのが State Chart[7]であり，OMT の状態図(以下，階層 STD と呼ぶ)は，これに基づいている．階層 STD は，状態遷移図のベン図表現である．スーパー状態に包含される状態は，スーパー状態の遷移を継承する．このことによって遷移を表す矢線の数を減少させ，また割り込みを自然に表現できる．動的モデルを強調することにより，オブジェクトの振る舞いの機能と制御を明確に分離できる．このことはメンタルモデルを運用

する際の，個々の認知判断とその実行順序の区別と一致している．とくに，意識の切り替えや，メンタルモデルでの意識の階層性や飛び越しを表現するのに適している．

4.2 関連の強調

リンクはオブジェクトインスタンス間の物理的あるいは概念的な結合である．関連は共通の構造と共通の意味をもったリンクのグループを記述している．いくつかの情報は本質的に単一のクラスを超越している．こういう情報が，単一のクラスに属するのではなく複数のクラスに依存していることを示すために，それらを関連としてモデル化する．メッセージの送受はリンクとして表現される．

人の行動は，場所や人間関係，相手の様子などによって行動に影響を受け，個人に解消できないことが多々ある．この点を表現するのに関連は便利である．

5. 分析結果

5.1 開発の指針

これまでの成果を“もの”とその関連という視点でオブジェクトモデルにし，外部の出来事による内部状態の変化を動的モデルで表現し，計算によるデータの変換を機能モデルとした．その図を基に，HFC と共に分析を進め，その結果を設計図に落とし，実装した．

システムの内容は，認知・判断そのものと各種情報の交換および取捨選択に分かれる．認知判断とそれに基づく手順の選択・実施は，プラントの構造に依存しているので本稿では詳説せず，情報に関する記述を中心とする．

情報の交換が分析の中心となるので，まずその一例を事象トレース図で示す(図 7)．これは先述した復水回収弁が開くという誤動作をマルファンクションとするシナリオである．運転チームは，原子炉運転員，タービン運転員と運転責任者である当直長から構成される．このシナリオでは，まず警報が発生し，それが全員に伝わる．警報の位置から，反射的に自分の担当であると判断したタービン運転員が見に行き，復水器ホットウェル水位低の警報が発生したことを当直長に報告する．この報告は原子炉運転員も傍受する．この警報情報をきっかけとして状況理解が始まり，主要状態量を定め，緊急性を判断するためにホットウェル水位という状態量を見る．状況理解の結果，緊急対応が必要であると判断され，阻止目標を定めて対応策を案出するために，主要状態量を確認する．続いてタービン運転員から復水補給弁を開くという提案がされ，協調過程にはいるが，当直長も同じ意見だったので了解した．そこで対応策の実施を開始し操作を行う．その後，また警報が発生し，いくつかの会話・動作が続く．

このようなシナリオとモデル化の前提を考慮して，作成したオブジェクトモデルを次節で述べる．

5.2 オブジェクトモデル

オブジェクトモデルのクラスは，文献[1]でのマイクロモデルにほぼ相当する．以下クラス名を「」で囲んで紹介する．

大きくは操作対象としての「プラント」と複数の「運転員」から成る運転チーム，両者を結ぶ「MMI」(man-machine interface) で構成される．「運転員」間のコミュ

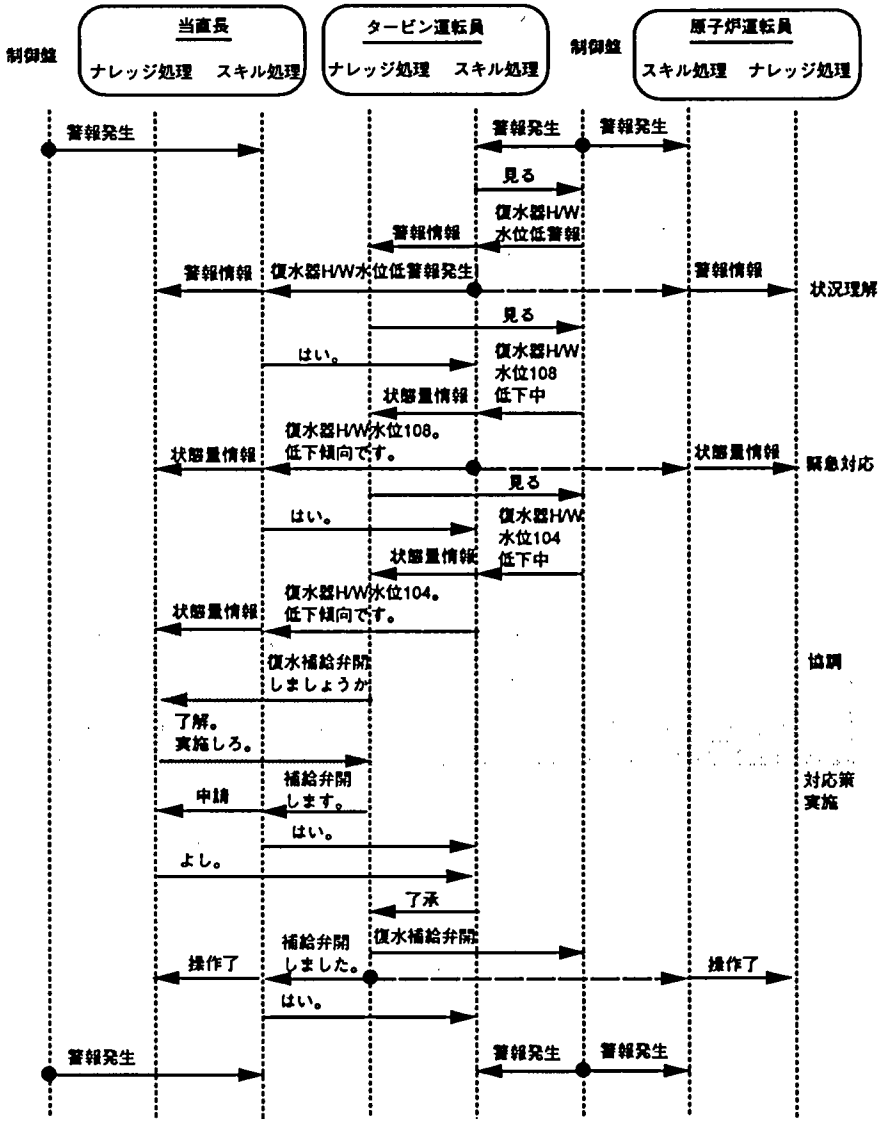


図 7　事象トレース図

ニケーションは,「HHI」(human-human interface)を通して行う.「運転員」は,「HHI」からの発話情報および「MMI」からの盤情報を入力とし，出力として発話や動作を行う.「運転員」は,「短期記憶」「中期記憶」「長期記憶」の記憶構造と「注意力」「思考」「動作」「発話」から構成される.「注意力」は個人内差を表現し,「思考」は認知判断を行い，思考の結果得られた発話意図や動作意図を「発話」「動作」が実行する．動作は「MMI」に対して行われ，発話は「HHI」を通して，相手の「注意力」に伝達される.「長期記憶」は，メンタルモデルの構築を担当し,「中期記憶」はその運用を行う.「短期記憶」は「注意力」を通過した情報を時間や量などの制限の下で保持する.

先述したように，分析の中心は情報の取捨選択である．これは情報過負荷(informa-

tion overload)の問題と密着し，またその解決が各クラスに分散しているので，先に概説する．

情報過負荷は，入力が受け手の処理能力を超えた時に起こる．個人の情報処理能力は変化していくが，これは個人内差として「注意力」で実現される．一般的には，情報過負荷への対応には，入力を無視する，処理の精度を下げる，待ち行列に入れ後で処理する，情報をふるい分ける，タスクを他へ振り分ける，タスクの実行を断念するなどがある．これらの中から対応を選択する基準は，情報・タスクの重要性や緊急性の判断による．これはシステムの前提に依存し，次のようにまとめることができる．

まず，チームとしての意思決定の鍵は，インデックスで処理されたメンタルモデルにある．そこで警報と操作に関する情報は聞きもらさないという前提をおく．誤警報・誤操作もないものとする．次に情報受取から発話・動作意図の形成までを思考の分割できない単位とし，この間は情報による割り込みは許さない．また，対応策の提案から決定までの協調過程も分割できない単位とする．このように思考中と協調中は新情報を処理できない．しかし，重要な情報を聞きもらしてはいけないので，この間に到着した情報は待ち行列に入れ1単位の処理が終了したところで，順次処理する．思考・協調は異常事象の進行に比べ短い時間で終了するので，チーム行動の模擬には差し障りない．ところで個人の「注意力」に到着した情報は，情報の注意喚起度によりスクリーニングされる．そして「注意力」を通過した情報は，「短期記憶」「スキル処理」を経て「状況理解」へ送られる．重要情報は注意喚起度を高くし，常に受け取れるようにする．情報の重要性は「状況理解」で吟味されるが，その基準は警報と主要状態量の変化である．重要でない情報はその状況では使用されないが，後続の処理で使用

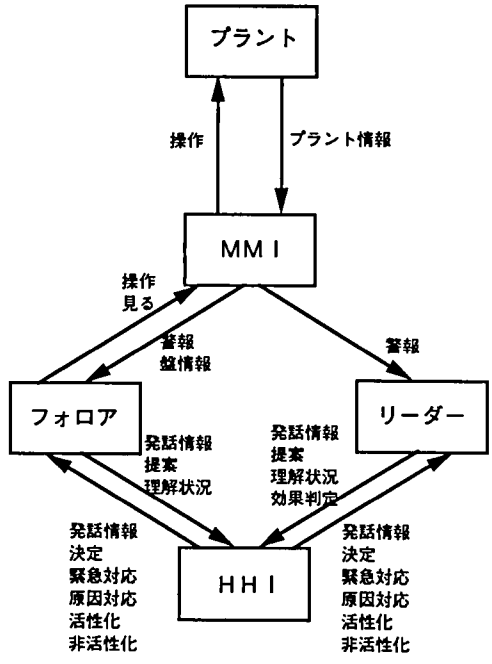


図 8　モジュール「運転行動チーム」の事象フロー図

できるように中期記憶に保存される．残るタスクの振り分け，タスク続行の可否は，チームの状態およびタスクの重要性に依存するので「HHI」が決定する．

メッセージの事象フロー図(図8)を示す．これを基に構成したオブジェクトモデルは，運転チーム行動モデル(図9)と長期記憶という二つのモジュールに分かれる．以下の節では各クラスの内容を述べる．

5.3 モジュール運転チーム行動モデル

5.3.1 運転員とチーム

運転員には，「リーダ」と「フォロア」がいる．「リーダ」は最終的な意思決定者としての運転責任者であり，「フォロア」は監視・操作の実施者である．「リーダ」のインスタンスとして当直長，「フォロア」のインスタンスとして原子炉オペレータとタービンオペレータを想定している．それぞれ異なる職分を持つ．運転員間のコミュニケーションを抽象化したものが，「HHI」である（図9）．

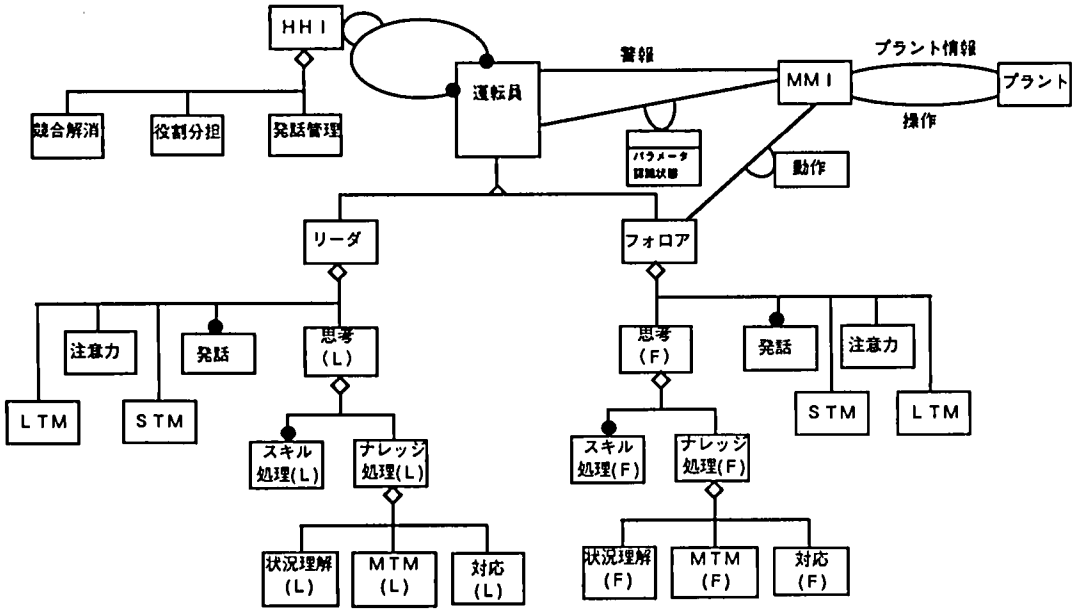


図 9　モジュール「運転行動チーム」のオブジェクト図

直接観察できるコミュニケーションは発話情報である．その内容は，プラントに関する情報の伝達，役割の通知，対応策の提案・指示とその諾否，実施した操作の報告などである．また他の個人と協調して意思決定する場合は，個性や声の大きさ・トーン，人間関係がその結果に影響する．これを行う「HHI」の機能は，発話管理・役割分担および競合解消である．「HHI」は高度な機能を持つが，ここでは個人モデルに影響する範囲で述べる．

発話管理では，個々の発話を然るべき相手にボリュームを計算して渡す．発話は基本的に「リーダ」と「フォロア」の二者間で行われる．ただしメンタルモデル形成上

の重要情報は，第三者にも伝える．また現実の世界では，相手の忙しさや，物理的位置関係，ことの重要性などを勘案して発話する．これを模擬するため，各個人から渡された発話文は「HHI」で相手の状況を判断し，発話の可否を決定する．

役割分担では，それぞれの運転員が対応として何を実施するかを，インデックス優先度や緊急性を加味して役割を分担する．これも各人の作業状況，盤との位置関係，職分などを考慮する．

競合解消では，運転員間で対応策が一致しない場合，どの対応策を採用するかについて，人間関係に依存した協調を図る部分である．これは対応策の提案があった時点で起動され，相手の案を尋ね，どちらかの案を対応策として決定し，両者に伝える．案が競合した場合は，提案に対する確信度や相手へ信頼感などを考慮して，討論の末に決定する．なお，先述したように協調中の運転員は，発話や盤情報は全て受け付けない．

さて，「リーダ」と「フォロア」は，個人特性などの属性が共通している．振る舞いもほぼ共通しているが，フォロアのみが動作することによる相違がある．そこで「リーダ」と「フォロア」は別クラスとし，その属性を汎化したものとして「運転員」をクラスとした．「HHI」は，個人に還元できない，チームとしての行動を明示するため「運転員」間の関連としてクラス化する．しかし「HHI」の各機能ごとにリンクの特性が異なる．役割分担はフォロア間の問題であり，競合解消は「リーダ」と「フォロア」の関係である．また，発話管理は基本的に「リーダ」・「フォロア」間の発話を扱い，「フォロア」の発話の一部を他のフォロアに傍受させる．これらのリンクを峻別し精緻に書くことより「HHI」のモジュール性を重視して，一つのクラスとし，「運転員」の関連とした．OMT の記法では 3 項以上の関係はモデルの煩雑さを増すことも，この選択の一要因である．「HHI」の各機能は，異質性を強調するために別々のクラスとした．

5.3.2 運転員とプラント

プラントは本来，連続的に変化する系であるが，このシステムでは一定間隔で実行を繰り返す離散的なものとして扱う．プラントと運転員を媒介するのが「MMI」である（図 9）．

「MMI」はそもそも，制御盤をモデル化したものである．しかしすぐ後に見るように，このクラスの機能はそれに留まらないので，ここでは二つの言葉を峻別する．

制御盤の持つ情報は，計器の値・警報発生の有無および操作具の ON/OFF や操作量である．計器の値および警報発生の有無は「プラント」から刻々と伝えられる．運転員と制御盤のやりとりは次の通りである．警報が発生すると意思とは無関係に各運転員に伝わる．そして運転員は目標情報を見ることによって，制御盤の情報を獲得する．また，操作することによって制御盤からプラントへ操作信号が送られる．

この限りでは，「MMI」は制御盤である．しかし以下のような点を考えると「MMI」は運転員とより密接な関係を持つ．まず，目標情報を見たとき，その周辺にある情報の変化に気がつくということである．ここでは，第一に運転員の視野は目標情報の盤上の位置と運転員の場所の関数であることと，第二に情報が変化したか否かはその情報と各人の認識のズレの度合いによって決まることが問題になる．次に，警報が発生した時にその警報を誰が見に行くかということである．これは警報の位置と運転員の

職分，場所，忙しさの度合いによる．

これらの問題は次のように処理する．運転員の位置や分担は，「HHI」と協同して判断する．また，周辺情報の変化はプラント状態に関する個々人の認識（パラメータ認識状態）と現在の制御盤の状態の差で判断する．その差が有意であるか否かは個々設定された閾値で決定する．パラメータ認識状態は，制御盤を見ることによっても，報告を聞くことによっても変化する．

このように「MMI」が抽象するものは複雑である．まず盤そのものがある．何に気がつき，何を記憶するかは盤と人間の関係である．また，盤の状態をどう認識するかは基本的に運転員と盤との関係であるが，盤の状態を他の運転員の報告によっても知り得るので，運転員と運転員との関係が影響する．さらに誰が操作するか，あるいは見るかは，盤と人間の位置関係そしてチームの特性による．この多項関係を精緻に描くことは，今回のモデル化の主眼ではない．そこでまず，「MMI」は盤そのものであるとし，パラメータ認識状態を「運転員」と「MMI」とのリンク属性とした．その上で，「MMI」はパラメータ認識状態を管理し，それと盤情報の比較により情報を選択的に送信する．送信データには，情報の種類や目立ち度を含める．また，送り先の運転員が情報を受け取れる状態であることを「HHI」に確かめた上で，送信する．

5.3.3 個　　人

「リーダ」および「フォロア」はほとんど共通の構造を持つので，以下では区別せずに叙述する．先に述べたように，「注意力」は個人内差を表現し，意図形成と実行を分離し「思考」と「発話」「動作」を別クラスとした（図10）．

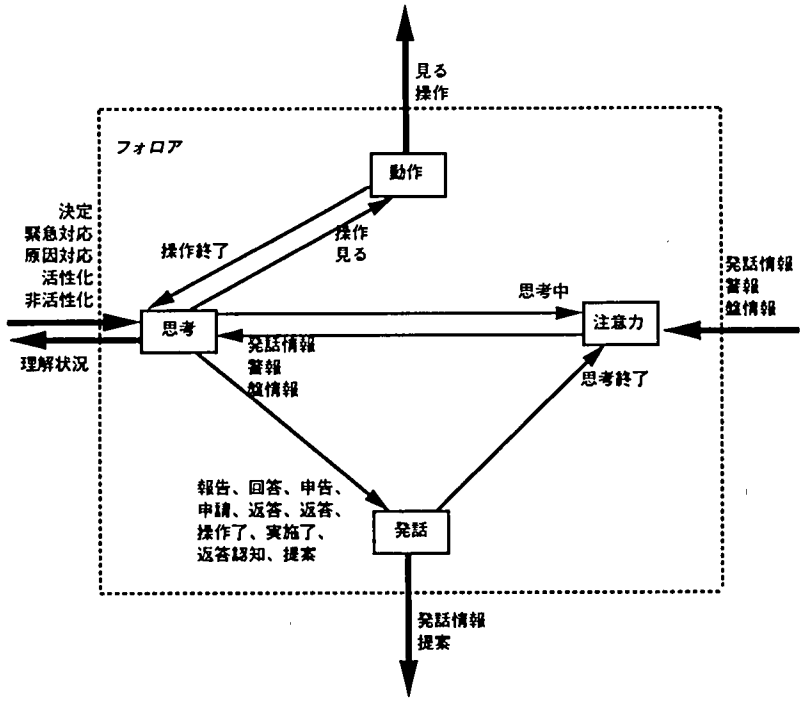


図 10　クラス「運転員」の事象フロー図

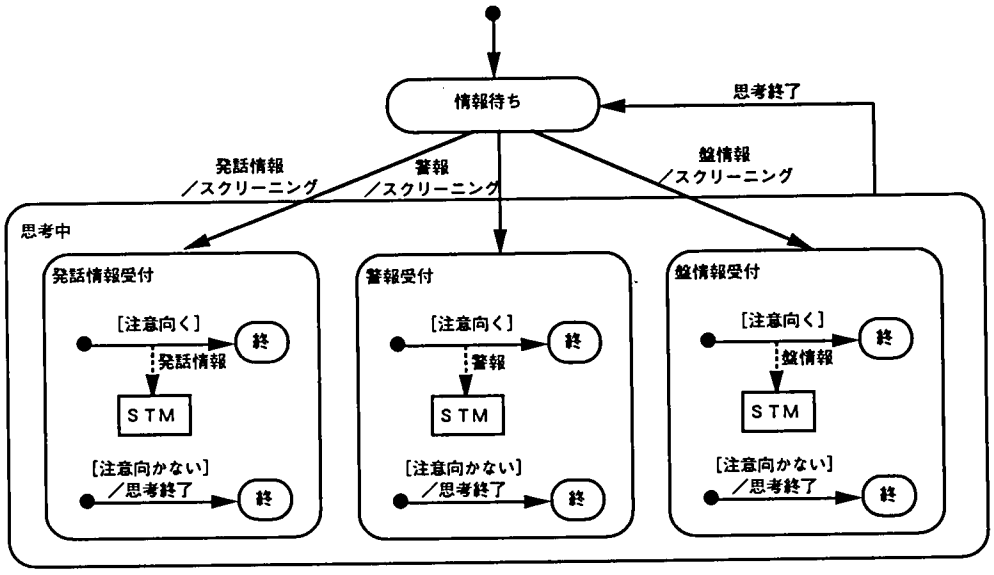


図 11　クラス「注意力」の状態図

1)　注 意 力……オペレータへの入力情報は「注意力」で受け付ける．受け付けた情報は現在認識できるかどうかをスクリーニングする．発話情報の場合，発話のボリュームと相手の職位によってスクリーニングする．盤情報の場合，計器類の注意喚起度，視野の範囲，注意力レベルでスクリーニングされる．注意力レベルは，覚醒度，精神作業負荷で決まる．認識した情報は，瞬時に短期記憶に伝えられる（図 11）．

先述したように思考中の場合は，情報を取り込むことはできない．これは，思考中に情報を取り込むとその情報により現在の思考を中断する可能性があり，中断した時点までの思考の扱いが難しくなるからである．逆に情報を取り込めない状況が多いと，シミュレーションは現実的でなくなる．そこで，分割し得ない思考の単位を次のように決めた．スクリーニングの結果，情報に注意が向かない場合や，注意が向いた場合に思考の結果として発話・動作意図が形成された時点で，思考は終了する．こうすると一回の思考時間は問題が生じない範囲に収まる．また思考中に到着した情報は待ち行列に入り，思考終了となった時点で次々と取り出す．スクリーニングは情報が持つ注意喚起度と作業者の覚醒度，作業負荷，性格などを加味したニューラルネットを利用して実現する．

2)　思　　考……認識情報を基にメンタルモデルの作成更新を行い，現在の状況を理解し，異常事象の推移の予測と原因の想定をして行動目標を決める．そして，発話・動作意図を形成し，「発話」・「動作」に伝達する．それゆえ，「注意力」のところで述べたように，基本的には発話意図・動作意図の形成によって思考の１単位は終了する．詳しくは次節で述べる．

3)　動　　作……見る・操作するといった動作意図を具体化し，動作時間を求めて実行する．動作時間は，標準動作時間にストレスや性格を加味して求める．動作

の実行は動作内容を「MMI」へ伝えることによってなされる．見た結果は盤情報として「注意力」に戻ってくる．

「動作」は運転員と制御盤との相互作用なので，「フォロア」と「MMI」の関連とした．

4) 発　　話……発話意図を具体的な言葉にし，発話時間を求めて発話する．発話の受け手は聞き取った場合，"はい"と返答し聞いたことを相手に伝える．一定時間後，返答がない場合は発話を繰り返す．

発話は一時に複数発生する可能性があり，それぞれ識別できるので個々の発話を一つのインスタンスとする．

5.3.4 思　　考

「思考」は大きく，「スキル処理」と「ナレッジ処理」に分かれる．「スキル処理」では，あらかじめ決められた通りの対応を実施する．「ナレッジ処理」では，得られた情報を基にメンタルモデルを形成し，それに従って，情報を整理して中期記憶に格納するなどの状況理解を行う．また，理解した状況に従って，緊急対応・原因対応など「HHI」で役割分担された対応を実施する（図12）．

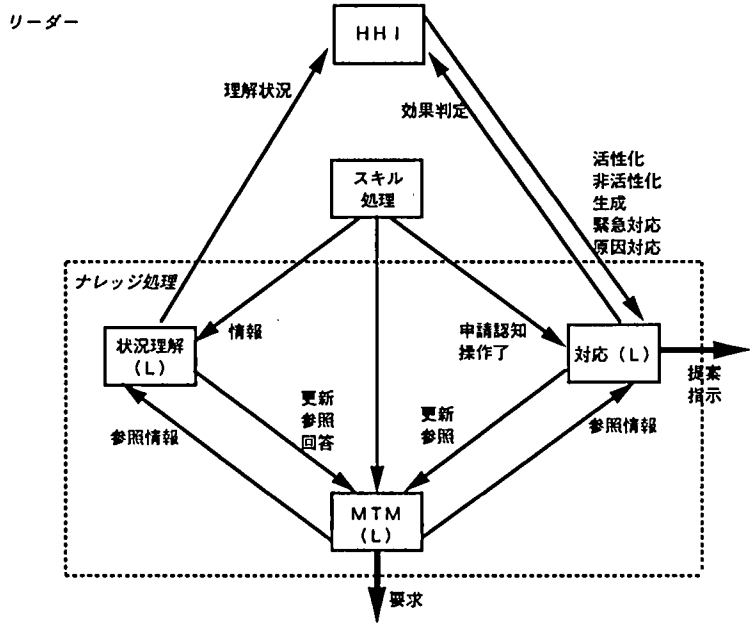


図 12　クラス「思考」の事象フロー図

1) スキル処理……獲得した情報がスキルベースで処理できるならば，ここで処理を完結し，そうでない場合は「ナレッジ処理」に任せる．スキルベースで完結する処理は，返答をすること，返答が返ってきた時その発話を終了すること，自分が担当の警報を見に行き，当面の対応をすることである．

2) 状況理解……状況の変化による人間の振る舞いを模擬するために，最も優先度の高いインデックスおよびそのメンタルモデルに対応する主要状態量が何かを監視する（図13）．

基本的な流れは，警報発生時のもので，まずメンタルモデルを作成あるいは更

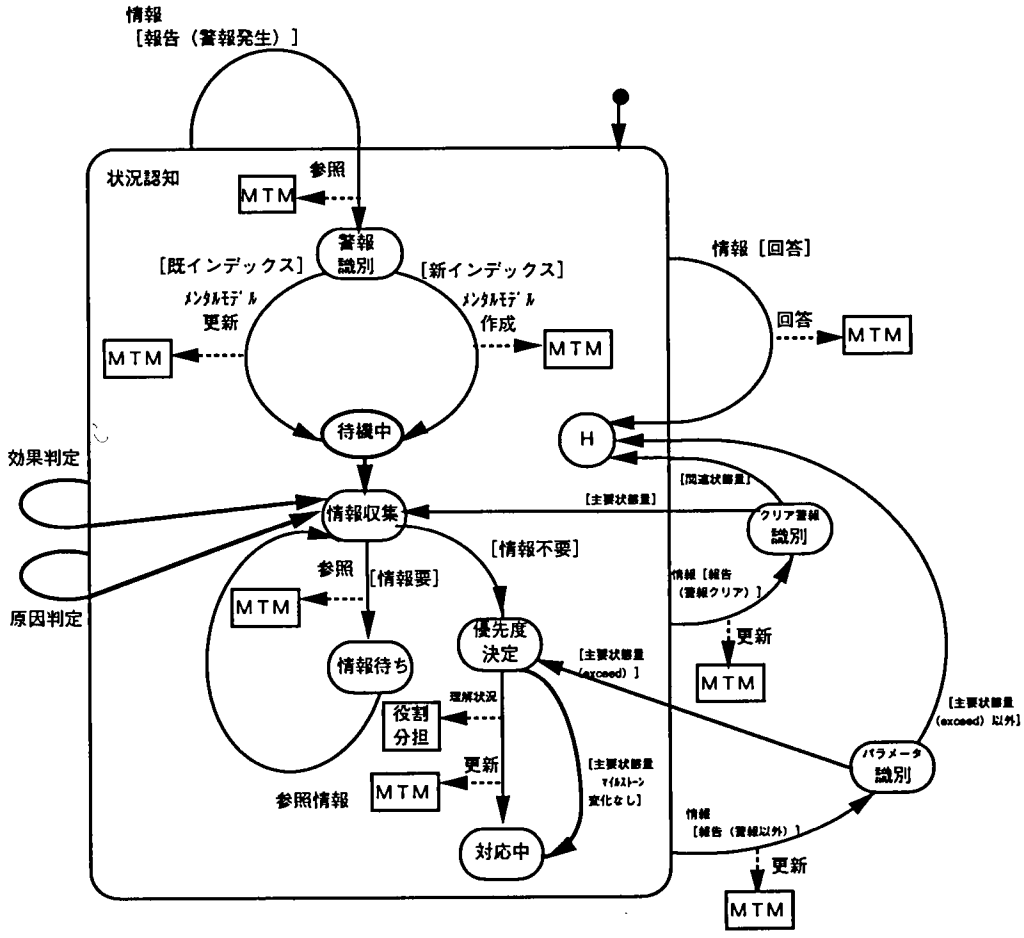


図 13　クラス「状況理解」の状態図

新し，必要な状態量について情報収集し，優先度を考慮して主要状態量を決定して「HHI」の役割分担に通知する．

「対応」で効果や原因が判定した時や主要状態量の警報がクリアされた時は，現在の主要状態量に関する処理が一段落したことになり，主要状態量の再考のために情報収集を始める．主要状態量が現在のマイルストーン（状態量が一定の値に達したら何かの対応策を講じようと決めている目標値）を超えた場合は，優先度を再計算する．

主要でない状態量に関する情報が来た時は，中期記憶にその情報を記憶し，割り込み前の処理を続行する．図中の H は状態履歴(History)を表し，H への矢印はスーパー状態の中で最近居た状態への遷移を意味する．

これらの処理は，中断・続行・やり直しを含み，状態履歴を含む階層 STD が有効である．

3)　対　　応……対応にはスーパー状態として緊急対応・原因対応・対応策実施・効果監視がある（図 14）．対応策案を決定するまでの過程が緊急対応・原因対応であり，いずれを行うかは「HHI」（役割分担）によって決まる．対応策案を提案し「HHI」（競合解消）によって決められた対応策を行うのが，対応策実施であり，

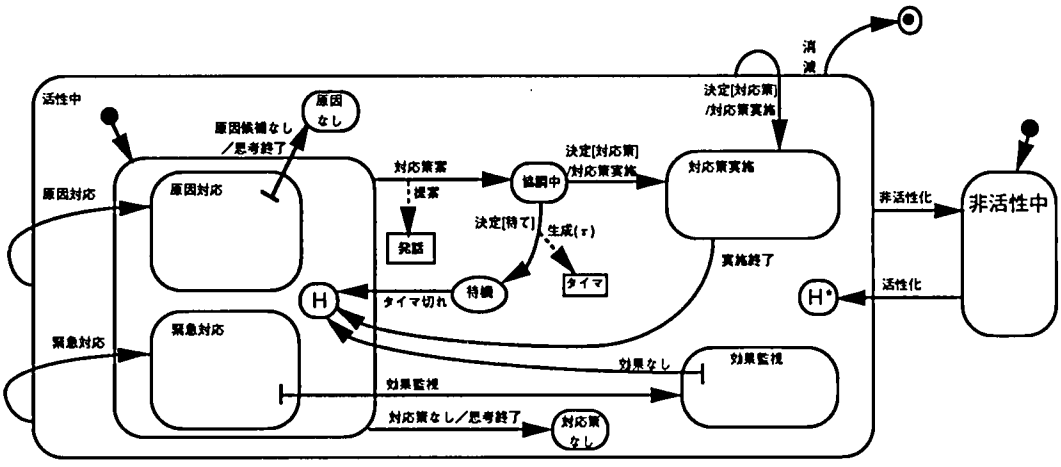


図 14　クラス「対応」の状態図

その効果を見るのが効果監視である．

個々の対応は，メンタルモデルの生成・更新プロセスで，インデックスに応じて生成され，主要状態量の交代・マイルストーン超過で白紙に戻る．そのインデックスに属する状態量が主要状態量である間は活性中である．ただし対応策を終了し，主要状態量が他へ移った後も，効果が判定されるまでは監視を続ける．効果監視は，監視する間隔と制限時間が設定されている．間隔毎に間欠的に監視し，制限時間を越えてもトレンドが変わらない場合は効果無しとし，次の対応策を考える．また，対応策を立案できない場合や，原因候補を列挙できない場合は，それ以上思考を進めることができないので思考終了となる．

なお図中の H*は，その下位状態に対しても状態履歴への遷移を再帰的に適用することを意味する．

5.3.5 記 憶 構 造

「短期記憶」は，「注意力」と「思考」をつなぐパイプである．ここを通る情報は「スキル処理」に伝達され，「MMI」に関する情報はパラメータ認識状態として記憶される．

「中期記憶」は，「思考」に取り入れた情報の内容を識別し，整理保存する機能と，「長期記憶」から取り出したメンタルモデルを運用する機能をもつ．情報の問い合わせに対しては，情報の存否と新鮮度を見て記憶内容から応答する，あるいは要求・操作するということを決める．

「長期記憶」は，プラント知識，一般知識，操作知識，経験知識をもつ．プラント知識はプラントの構造，プラントパラメータがもつ値の意味と相互の関連およびプラント全体の挙動に関する知識である．一般知識は，言葉の理解や用語に関する知識である．操作知識は，操作手順に関する知識である．経験知識は，これまでの経験についてのエピソード記憶を整理して記憶している．

5.4 モジュール長期記憶

先に述べたようにメンタルモデルの装置のトポロジ・想見・因果モデルを担当するのが長期記憶の役割であるが，今回その一部分である因果モデルをプラント知識とし

て持つ．

そこで知識ベース(Knowledge base；以下KB)は，異常事象の予測シナリオをたてるのに必要な情報を中心とする．プラント知識は，インデックスごとに整理された異常事象と状態量の関連およびそれに付随する原因，対応策などの関係で表現される(図15)．各運転員の長期記憶には，これらの部分集合に対応するKBを持つ．長期記憶のKBは，誰もが共通に持つ知識と個別に内容が違う知識に分かれる．チームとしての行動が成立するには，①相手の言っている単語が理解できる，②言われた対応策を実行することができる，③相手の考えの根拠が理解できるといった条件が必要である．運転員は皆訓練を受けていること，議論は対応策の実施のためのものであることから，現在得られている知識・「HHI」の協調方式から緊急対応策とマイルストーンだけを個人別のKBとし，それ以外は共通KBとした．

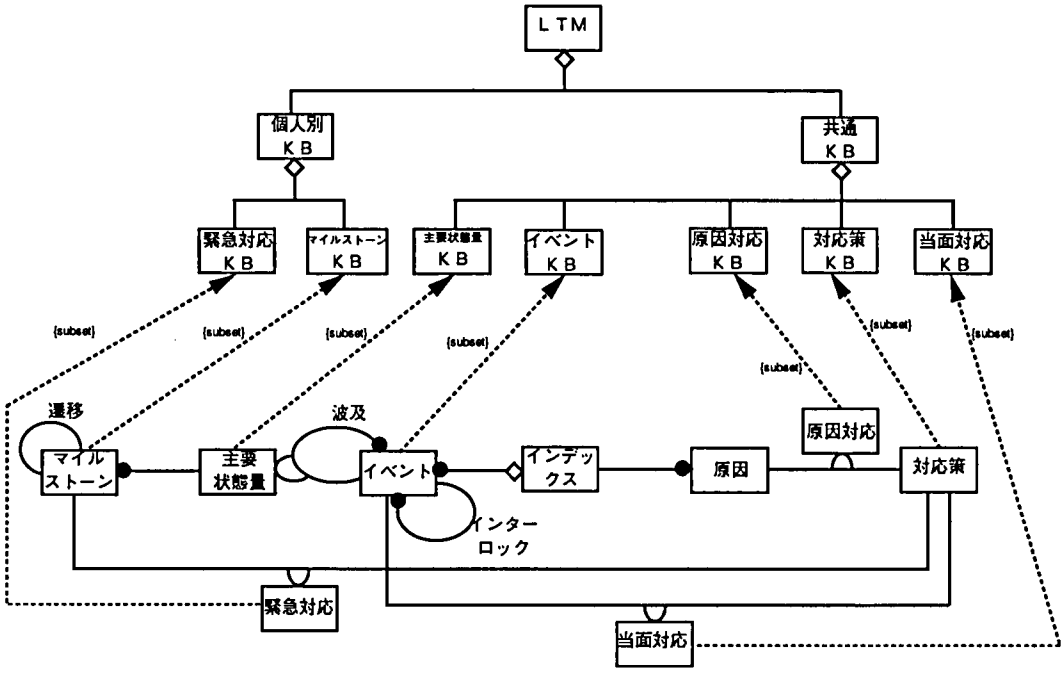


図 15　モジュール「長期記憶」のオブジェクト図

5.5　整合性について

各インスタンスはすべて並列に動き，他のオブジェクトとメッセージをやりとりしながら動くため，一つあるいは複数のオブジェクトが実行を続けられない状態になる可能性がある．今回は「HHI」の機能に基づく要請として，次の二つを前提とする．

① 誰も警報および警報情報を聞きもらすことはない．

② 役割分担，競合解消への依頼は必ず解決される．

これを前提にして，以下の点を検討した．

ⓐ 思考は必ず終了するか．

ⓑ 全員の注意力が高く，スクリーニングの結果，必ず注意が向く場合は停止することはないか．

分析モデルに基づく検討の結果，非活性中の「対応」にメッセージが送られた場合，思考が終了せず停止してしまうことが発見された．これは現実の世界では，主要な関心事でない，あるいは関心の無くなった事柄について言われても，発話意図や動作意図は生じないということである．ところが，「対応」にメッセージを送信するのは，個人の「スキル処理」であるが，「対応」の活性/非活性を制御しているのは「HHI」であり，個人の中では対処できない構造になっている．そこで，「HHI」の発話管理機能において，非活性中の対応に対する発話は，相手に届けないという機能を追加することで回避した．これは，前提と矛盾しないので安全である．

6. 設計・実装の方針

設計では，操作や属性のオブジェクト図へのマージとシステムの全体的構成とその制御方式を決定する．

6.1 システムアーキテクチャの選択

プロセスとしてモデル化とプログラミングは明確に分離されているが，そのモデル表現は若干の前提を置けば素直にプログラムへと変換できる．とくにこのシステムは，シミュレーションシステムなので，オブジェクト指向と親和性が高い．OMTの教科書[2]では共通アーキテクチャの一つとして，動的シミュレーションを挙げ，オブジェクト指向アプローチを使って最も容易に設計できるシステムとしている．そして，FSMをシミュレートするコントローラと複数オブジェクト間のメッセージ交換メカニズムを導入して制御を実装し，オブジェクトと操作をアプリケーションから取り出せばよい，としている．今回の開発では，この方針を踏襲しシステムアーキクチャをシミュレーションシステムと定め，C++とそのタスクライブラリで実装した．オブジェクト設計と実装はこの前提をおけば素直に実現できるので，省略する．シミュレーションとOMT，C++およびFSMの関係を以下に述べる．

6.2 シミュレーションとOMT

まず離散系シミュレーションについて概観する[9]．離散シミュレーションは，時間の進行に伴うシステム状態の変化を模擬する．システムの状態の変化は，システムの中に存在する”もの”の変化である．”もの”を構成要素(entity)と呼び，構成要素はその性質を表す属性(attribute)を持つ．また，構成要素は集合(set)をなすこともある．モデルの存在する構成要素とその属性の値によって，システムの状態が確定する．したがってモデル内の構成要素の変化，属性の値の変化，集合の要素の変化が状態の変化となる．

システムの状態に何らかの変化をもたらす出来事を事象と呼ぶ．この変化は，時間の進行を伴わず，瞬間的に起こると仮定する．つまり，状態の変化を連続的にはとらえない．事象には，他の事象とは無関係にある時間間隔に起きるものと，他の事象が起きて状態が変化したために引き起こされるものとがある．事象の発生にともなう，状態の変化の仕方を記述したものを事象ルーチン(event routine)と呼ぶ．一つのサービスを活動(activity)ととらえる方式もあるが，状態の変化を連続的にとらえないという仮定から，活動はサービスの開始と終了に分離できる．これは同一モデルの異なる記述方法である．そして時間の進行方法は，次の変化が起きる時刻まで時間(simula-

tion clock)を進めて状態の変化を記録し，また次の変化の起きる時刻まで時間を進めるという，事象—事象進行方式をとる．現段階でのシミュレーション言語は，逐次的な実行を仮定しているので，同時刻に起こる事象は実際には全順序化されて実行される．この順序を同時刻事象間に優先順位を付けるなどの方法により，アプリケーション側から指定する機構を提供する言語もある．

OMTと対比すると，事象・状態の概念はそのまま動的モデルのそれに対応し，エンティティは基本的にオブジェクトに対応する．事象優先度や高階の割り込みなどより細かい指定は，関連やSTDの階層性を用いて記述できる．ところでOMTにおける時間は，事象の前後関係としてしか表現されない．実時間と対応させるには，StateChartにあるようなタイマ機構を導入する必要がある[8]．

6.3 シミュレーションとC++

これまで述べた概念にほとんど素直に対応したシミュレーション用プログラミング言語がSimscript II．5であり，状態変化を活動としてとらえたのがSIMULA67である．Simula67は，抽象データ型機能をクラスとして実現しているという特徴を持っている．クラスを型紙とした"もの"が動く姿をコルーチン機能を用いて実現している[10]．C++はSimula 67から取り入れたクラス概念を提供しており，残りのコルーチン機能は，クラスライブラリの一つとして，タスクライブラリの名で提供されている．シミュレーション言語の仕組みとの関係でいうと，事象ルーチンはタスクとなる．構成要素は，インスタンスであり，その集合がクラスである．属性は，クラスごとに定義された各インスタンスの変数となる．状態の変化は，タスクの活動として定義する．タスクは活動を始めると何らかの動作をし，活動終了後は再開するきっかけを宣言して休止状態となる．このきっかけは，事象に対応する．このライブラリでは，時間を疑似時間(simulated time)として扱える．また，集合操作の一種として，待ち行列を操作する機能も提供されている．

制御メカニズムのうち，メッセージ交換に基づく複数オブジェクトの実行は，タスクライブラリのコルーチン機構と待ち行列機能を利用できる．FSMをシミュレートする機構は無いので，自作した．同時性など時間の扱いは，コルーチン機構と待ち行列機能に従った．

ライブラリの制約からタスクは階層を持てない，またC++にクラス操作がないことから，クラスの集約・継承やインスタンスを表現する独自のメカニズムを作った．

6.4 状態機械エンジン

FSMは有限個の状態を持つ仮想的な機械であり，状態は機械の過去の事象履歴をまとめた表現である．これはこの機械が，その時の入力とそれ以前の入力に依存した出力をする，記憶を持つ機械であることを仮定している．オブジェクトは，一つのメッセージを入力されるといくつかのメッセージを出力する機械と見なせる．FSMのモデルにはMealy型とMoore型があり，前者は，出力が遷移に伴い，後者は状態に伴う．これは，シミュレーションにおける事象中心型と活動中心型に対応する．このシステムは割り込みの表現が重要であり，また事象中心型で素直に設計できるため，Mealy型を採用した．するとFSMの抽象モデルは表1のようになる．

表 1　有限状態機械のモデル

1	状態の有限集合	$S = \{s_0, s_1, s_2, \cdots\}$.
2	初期状態(initial state)	$s_0 \in S$.
3	入力事象の有限集合	$I = \{i_1, i_2, \cdots\}$.
4	出力事象の有限集合	$O = \{o_1, o_2, \cdots\}$.
5	状態遷移関数(transition function)	$f : S \times I \rightarrow S$.
6	出力関数	$g : S \times I \rightarrow O$.

これを文章で表すと次のようになる．任意の瞬間において FSM は，状態の一つに居る．そして，入力事象の到着で，FSM は状態遷移関数に従って別の状態へ移動する．状態遷移に付随して，出力事象を出力する．なお，最初，FSM は初期状態に居る．こうすると事象は，不連続な時点で発生していることになる．この事象の前後関係による半順序集合が時間の概念を生み出す．

OMT との対応では次のようになる．一つ一つの FSM は，個々のオブジェクトに対応する．一つのクラスに属するオブジェクトは STD を共有する．定義より個々の FSM は逐次プロセスとなり，タスクに対応できる．

以上の考察から，FSM とそのエンジンは図 16 のように設計できる[11]．これは，従来型の STD に状態の汎化を導入したものとなっている．

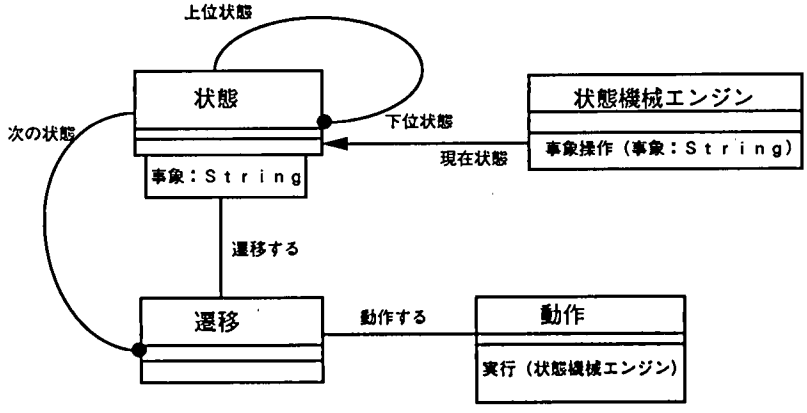


図 16　状態機械エンジンのオブジェクト図

さて，各 FSM に到着した事象は，到着順に消費される．そして，事象を消費しようとした時点で，現在状態に対応する遷移がない場合，その事象は何も起こさずに捨てられる．これでは，表現が不自然になる場合があるので，特別の状態として，活性中と非活性中を設けた．非活性中の場合は到着した事象は貯められ，活性化した時点で消費される．この他，状態履歴の機能を加えて，状態機械エンジンを実現した．

7. おわりに

SYBORG システムの開発をシミュレーションモデルと OMT 記述の関連を軸として述べた．全体としては，OMT 記述設計者・開発者・適用領域の専門家の共通する表現手段となり，モデルと表現が直接的に結びついた点で成功した．

シミュレーションシステムであるということから，オブジェクトモデルおよび動的

モデルを扱った．具体的な材料は発話なので，メッセージの交換を中心にみた．そのために，関連と階層STDの扱いが重要となった．関連を使うことによってオブジェクトに持たせる情報が明確になった．ただし3項以上の関連は2項関連に置き換えることで議論しやすいモデルとした．また階層STDは強力であり，文脈にそったメッセージ応答を描ける．しかし，人間のメッセージ応答は目標や周りの環境により影響され，過去の履歴だけが文脈を決めるわけではない．目標の決定は機能モデルで，環境は関連で表現できるが，その場合全体としての意味付けが見えにくくなるという弱点がある．

メッセージ交換に焦点を当て，機能モデルの記述を割愛したため，エキスパートシステム構築用言語を用いたメンタルモデルの構成，ニューロによる注意喚起度などの表現，発話文の形成と音声情報への変換，プラントモデルを含めたGUI構築ツールを利用したリアルタイムシミュレーションの仕組みについては，触れることができなかった．

参考文献

［1］ 吉村他，運転チーム行動モデルの開発，技報41号，5月号，1994.
［2］ J.Rumbaugh, et all, 羽生田　監訳，オブジェクト指向方法論OMT, 1992, トッパン
［3］ 佐相他，原子力発電所における運転員のチーム行動シミュレーション技術の開発(その1)，電中研研究報告，S 91001(1992).
［4］ 佐相他，運転チーム行動モデル検証用プラントシミュレーションコードの開発，電中研研究報告，S 92002(1993).
［5］ J.Kleer & J.S.Brown, 北上　訳，機械的メンタルモデルにおける仮定とあいまいさ，メンタルモデルと知識表現　所収，1986，共立.
［6］ J.Rasmussen, 海保他訳，インターフェースの認知工学，1990，啓学出版.
［7］ 高野他，運転チーム行動モデルの全体構成とオペレータ行動モデル（個人モデル）の設計，電中研研究報告，S 92001, 1993.
［8］ D.Harel, Statecharts: A Visiual Formalism For Complex Systems, Science of Computer Programming, 8, 1987.
［9］ 関根他，シミュレーション，日科技連，1976.
［10］ K.Nygaard, &D.J.Dahl, The Development of the Simula Language, ACM Conf.on History of Programming Languages.
［11］ J.Rumbaugh, Controlling code - How to implement dynamic models, Journal of Object Oriented Programming, 1993 May.
［12］ J.Rasmussen, "Skills, Rules, and Knowledge;Signals, Signs, and Symbols, and Other Distinctions in Human Performance Models", IEEE Trans.of Systems, Man, And Cybernetics, vol. SMC-13, no. 3, 1983.

執筆者紹介　羽　田　昭　裕（Akihiro Hada）

1984年一橋大学社会学部卒業．同年日本ユニシス（株）入社．意思決定支援ソフトウェアの開発・適用などに従事．現在 システム技術第一本部 研究開発室に所属．情報処理学会会員．

バッチラン運用管理システムにおける所要時間の予測とスケジュール作成

A Running Time Prediction Model for a Batch Processing Support System

松田芳雄, 河岸憲一

要 約 大規模システムのバッチ運用は，処理量の増大とオンライン時間の延長等により効率向上が課題となっている．今回バッチ運転時間の短縮化を目的としたバッチ運転スケジュール作成手法を検討し，またその手法を実機に適用し効果を確認した．

スケジュール作成のためにはバッチ処理時間の予測が必要である．そのために行列理論と反復手法を用いた処理時間の予測手法を開発した．次にランの開始時刻制御やデッドライン制御により計算機資源を有効に利用し，バッチ運転時間を最小化するスケジュール作成手法を開発した．

Abstract Stemming from by increasing data volumes and the extended running hours of on-line computer systems, a new challenge has been how to make batch data processing by large-scale information systems more efficient. The author and his team have recently explored a new way to schedule batch processing operation with the aim of reducing computing time for batch applications. The brainchild of the efforts was run on an actual system to see how it would work.

Such scheduling requires the process of predicting the time for batch data processing. That is why a new prediction method had to be developed through the use of the queuing theory and the iteration method. Then, on the heels of it, a new scheduling formula has been developed, which controls run-starting time and deadlines to minimize batch operation hours by making the most of computer resources.

1. はじめに

大規模システムのバッチ運用は，処理量の増大とオンライン時間の延長等により効率向上が求められている．通常，バッチ処理の運用は運用管理システム（たとえば，COSTAR(Computer Operating Management System for Total Automation & Reliance)）により管理され，あらかじめ決められた処理順序や指定時刻に従い，バッチランの自動起動などが行われている．オペレータはコンソール上のメッセージをもとに進捗状況を管理しているが，各ランの終了時刻の予測や臨時ランの投入時期の判断などは勘で行っている．このため，バッチランの進捗監視と終了予測，スケジュール作成を行うバッチランの運用支援システムが求められている．

バッチランの終了予測およびスケジュール作成のためには，バッチランの所要時間を推定することが必要になる．しかし，所要時間は処理するデータ量やランが走行する計算機環境により変動し，簡単に予測することはできない．

本稿では，UNISYS 2200/1100 シリーズ計算機における，バッチランの所要時間の予測方法について紹介する．さらに，最適スケジュール作成の方法について述べる．

2. バッチランの運用支援システム

バッチランを効率よく運用するためには，スケジュールの予測と作成機能を持つ以下のようなシステムが必要である．

1) スケジュール予測機能……運用管理システムから指示されたとおりにバッチランを走行させた場合，各ランのスケジュール（開始時刻と終了時刻）を予測する機能である．

① 朝，運用管理システムより当日走行予定のバッチランの一覧表を貰い，1日の走行スケジュールを予測する(図1)．さらに，CPU利用率など計算機資源の利用状況も予測する（図2)．

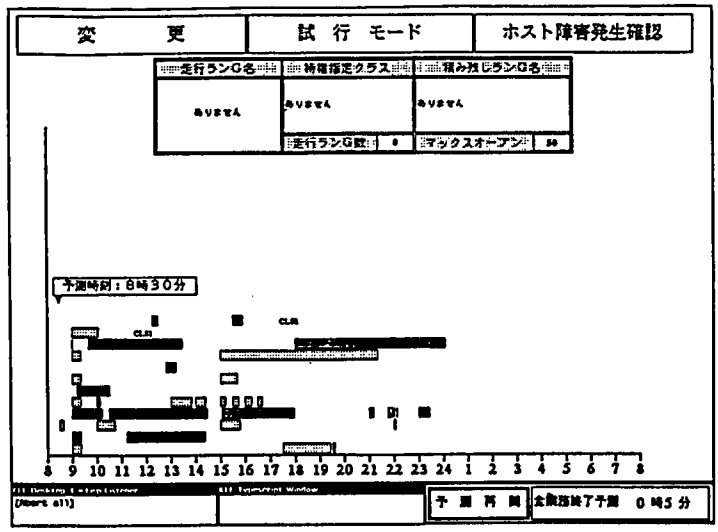


図 1　バッチランの業務別走行スケジュール

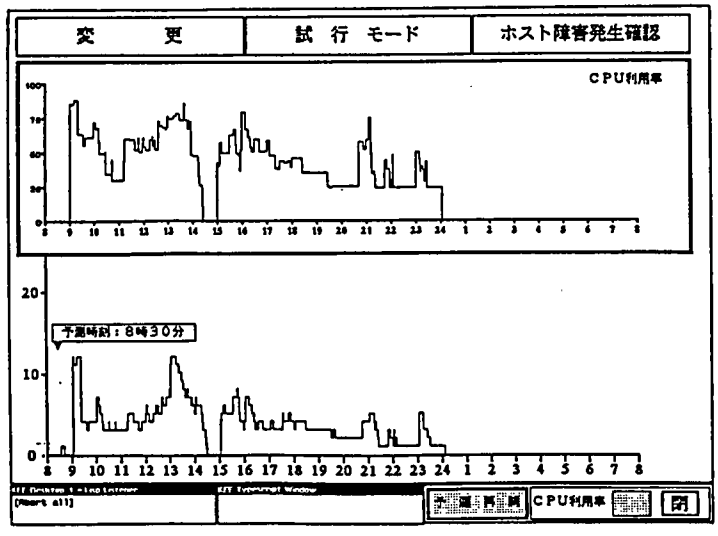


図 2　同時走行ラン数，CPU利用率の推移予測

② バッチ運転の途中において，ホスト計算機から各バッチランの終了状況を貰いながら，定期的にスケジュール予測をやり直す．

③ 走行予定ランの変更があったとき，スケジュール予測をやり直す．

④ ホスト計算機の障害時や縮退運転への切り替え時にスケジュール予測をやり直す．

⑤ 臨時ランが発生したとき，シミュレーションを行い適切な投入時刻を探す．

2) スケジュール作成機能……全体の終了時刻が最も早くなるようにバッチランのスケジュールを作成する機能である．スケジュール作成結果を運用管理システムに反映することにより，最適な自動運転が可能になる．

① 朝，運用管理システムより当日走行予定のバッチランの一覧表を貰い，1日の走行スケジュールを作成する．さらに，CPU 利用率など計算機資源の利用状況を予測する．

② 走行予定ランの変更時やホスト計算機の障害時にスケジュールを作成し直す．

③ その他，オペレータの指示がある都度スケジュールを作成し直す．

スケジュールの予測，作成のどちらにおいても，各バッチランの所要時間を予測することが重要な要件となる．以下にバッチランの所要時間の予測モデルについて述べる．

3. 所要時間予測モデル

3.1 予測モデル概要

バッチランの所要時間は，CPU および I/O 装置の使用時間とその待ち時間，メモリ待ち時間から構成される（式 3-1）．他にオペレータの応答待ちなどの要因も考えられるが，全体に占める割合が少ないことと，解析が困難であることから，ここでは考えないことにする．

$$\begin{aligned}(\text{所要時間}) = &(\text{CPU 使用時間}) + (\text{CPU 待ち時間}) \\ &+ (\text{I/O 使用時間}) + (\text{I/O 待ち時間}) \\ &+ (\text{メモリ待ち時間}) \end{aligned} \tag{3-1}$$

したがって，各バッチランの所要時間を予測するためには，それぞれのランの CPU，I/O 装置の使用時間と待ち時間，メモリ待ち時間を予測し，合計すれば良いことになる．

2200/1100 シリーズ計算機では，CPU，I/O 装置の使用時間は，正確には知ることはできないが，SUP（Standard Unit of Processing）値という計算上の値がこの代わりに使われている．SUP 値はバッチランの実行の度に一定ではなく，処理するデータ量により変動する．しかし，多くのランは月末や週末に処理量が集中したり，特定の日や曜日に集中したり，規則性がある場合が多い．そこで，この規則性を過去の稼働実績データをもとに統計的手法によりモデル化し，予測を行うことにする．

一方，待ち時間は，バッチランの走行時の環境（同時走行ラン数，CPU 対 I/O 比率，…）により変動するので，過去の実績から推定することは困難である．ここでは，各環境下における待ち時間を計算機シミュレーションにより事前に求めておき，バッチ

ランのSUP値と走行時の環境を考慮して所要時間を求めることにする．

以下に，使用時間予測モデル，待ち時間予測モデルおよび所要時間の推定方法について記述する．

3.2 使用時間予測モデル

1) SUP値

2200/1100シリーズ計算機の場合，SUP値が計算機使用時間として使われている．SUP値は，CPU，I/O，CC/ERの三つに分けて管理されている．CPU・SUP値はCPUの使用時間，I/O・SUP値はI/Oの使用時間の計算上の値である．CC/ER・SUP値はOSがそのバッチランのために処理した時間を表しており，CPU時間と若干のI/O時間を含むが，分離は不可能なのでCPU・SUP値に含めて考えることにする．

2) SUP値予測の考え方

各バッチランのSUP値は，実行の度に一定ではなく，処理するデータ量により変動する．処理データ量はバッチランの実行の直前には分かるので，それをもとに各ランのSUP値を予測することは可能である．しかし，走行するランの数が多くなると，データ量の入力だけでも大変な作業になる．また，当日の朝に予測を行う場合には，データ量が不明なことが多く，予測が行えないことになる．そこで，もっと取扱いが容易な要因からSUP値を予測する方法を考える必要がある．

経済活動や企業活動は暦に従って行われることが多いので，それから発生するデータも月，日，曜日等により変動している場合が多い．これらをモデル化し，SUP値を予測することを考える．

3) SUP値予測モデル

日次ランを例にSUP値予測モデルについて説明する．SUP値の変動要因として「月」，「日」(「日付」)，「曜日」を考え，i月j日k曜日のSUP値の予測値S_{ijk}を以下のように表すことにする．

$$S_{ijk}=\mu+M_i+D_j+W_k \qquad (3\text{-}2)$$

ここで，
- S_{ijk}：i月j日k曜日のSUP値の予測値
- μ：基本値
- M_i：i月のウェイト　　i = 1, 2, …, 12
- D_j：j日のウェイト　　j = 1, 2, …, 31
- W_k：k曜日のウェイト　　k =月,火,…,日

すなわち，SUP値の予測値は，基本値μに当該月のウェイトM_iと当該日のウェイトD_j，当該曜日のウェイトW_kを加えたものとして考える．基本値および各ウェイトは過去の実績データをもとに最小二乗法により推定する．

このモデルにより，あるバッチランのTOTAL・SUP値を予測した結果を図3に示す．横軸が実行日で縦軸がTOTAL・SUP値である．推定値と実績値の相関係数（重相関係数）は0.90で，簡単なモデルではあるが十分予測可能なことがわかる．CPU・SUP値とI/O・SUP値も同様に求めることができる．

4) 予測モデルの高精度化

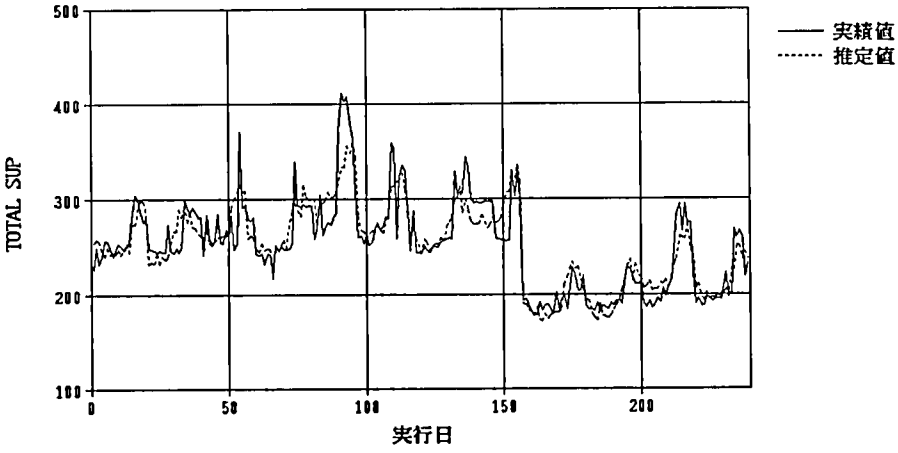


図 3　TOTAL・SUP 値の予測結果

式 3-2 のモデルでは，日次の変動を説明するために「日付」を用いたが，休日の関係で必ずしも日付と変動が一致するとは限らない．そのため，以下の二つの要因を導入する．

イ．回数 T_t …… 当該月の何回目の実行かを表す．

t = 1 回目，2 回目，…

ロ．最終回 L_l …… 当該月の最後の実行か否かを表す．

l = 最終回，最終 1 回前，その他

「日付」の代わりに「回数」や「最終回」を用いた方が，月内の周期変動をより精度良く近似でき，予測の精度も向上する．しかし，「回数」を用いた場合には当日の実行が当該月の何回目かを記録していなければならないし，最終回か否かの判断も必要になり，予測のための手続きが煩雑になる．予測の精度と操作性を考慮して説明要因を決める必要がある．

処理量の増大やデータの蓄積により，SUP 値が時系列的に増加する場合がある．このときには，以下のような傾向変動を表す変数を導入する．

イ．年月日 …… 一次の傾向を表す．最初の日を 1 として連続番号で表す．

ロ．年月日二乗 …… 二次の傾向を表す．(年月日二乗) = (年月日)^2

5)　予測モデルの運用

バッチラン運用支援システムにおいては，SUP 値予測モデルは以下のように運用・更新する．

①　以下の二つのデータベースを用意する．

イ．稼働ログデータ …… 各バッチランの過去の実行ごとの SUP 値を保存する．

ロ．予測モデルデータ …… 各バッチランの SUP 値予測のためのウェイト（式 3-2 の基本値と各ウェイト）を保存する．

②　予測当日，走行予定ランの SUP 値を予測モデルデータをもとに予測する．

③　実行が終了したランの実績 SUP 値を稼働ログデータの追加保存する．

④ 予測モデルデータのウェイトを稼働ログデータから定期的（毎日，毎週，毎月，半年ごと，…）に更新する．

3.3 待ち時間予測モデル

1) 待ち時間予測の考え方

バッチランの走行環境が異なれば待ち時間も異なる．したがって，待ち時間はSUP値のように過去の実績データから予測することはできない．しかし，走行環境と待ち時間の関係がわかれば待ち時間を予測することができる．ここでは，走行環境として同時走行ラン数とCPU比率（バッチラン全体のTOTAL・SUP値に対するCPU・SUP値の比率）を考える．そして，以下のような関係（式3-3，式3-4）を事前に求めておき予測を行うことにする．

(CPU待ち時間)＝f｛(同時走行ラン数)，(CPU比率)｝　(3-3)

(I/O待ち時間)＝g｛(同時走行ラン数)，(CPU比率)｝　(3-4)

(f，gは同時走行ラン数，CPU比率の関数を表す)

式3-3，式3-4の関係は計算機を用いた待ち合せ型シミュレーションにより求めることができる．

2) シミュレーション・モデル

計算機システムはCPU，メモリ，I/O装置などで構成されている．バッチランはこれらの装置に対し処理要求を出し，OSがこれを管理している．これらの装置やバッチラン，OSの動きをモデル化し，シミュレートすることにより，待ち時間を算出することができる．

システム内にNランのバッチランが存在するとする．NランのバッチランがCPUとI/O装置を使う比率をR：1-R（CPU比率，0<R<1）とする．このとき各装置での待ち時間を求めるために以下のようなモデルを作成する．

イ．各バッチランはR：1-Rの比率でCPUとI/O装置を使う．

ロ．CPUまたはI/O装置が空いていれば一定の処理を受ける．

ハ．空いていない時は待ち行列に並び，空くまで待って処理を受ける．

ニ．処理の終わったバッチランはイ.に戻り，上記を繰り返す．

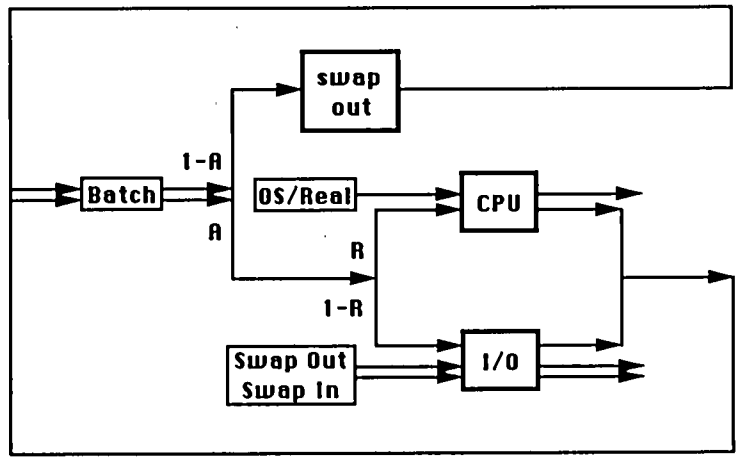
このモデルを計算機でシミュレーションすれば，CPUやI/O装置での待ち時間（処理要求1回当り待ち時間）を測定することができる．シミュレーションにはGPSS（General Purpose Systems Simulator）などの汎用のシミュレーション言語を使用する．

3) 走行ラン・モデル

モデル化の前提として，バッチ専用機や夜間バッチ業務を対象とすることを基本とし，計算機システム内にはバッチランとOSだけが走行しているものとする．OSにはシステム制御のためのリアルタイムランも含めて考える．2200/1100シリーズ計算機ではメモリ待ちはスワップアウト/インまたはページアウト/インとして現れるので，このためのI/Oも走行ランとして考える．すなわち，システム内にはバッチラン，OS，スワップアウト/インまたはページアウト/インの3種類のランが走行する．メモリ待ちをスワップアウト/インで対応するモデルをC型モデル，ページアウト/インで対応するモデルをM型モデルと呼ぶことにする．そ

れぞれのモデルで各走行ランは以下のように挙動するものとする．

【C型モデル】(図4)



A：インコア比率　　R：CPU比率

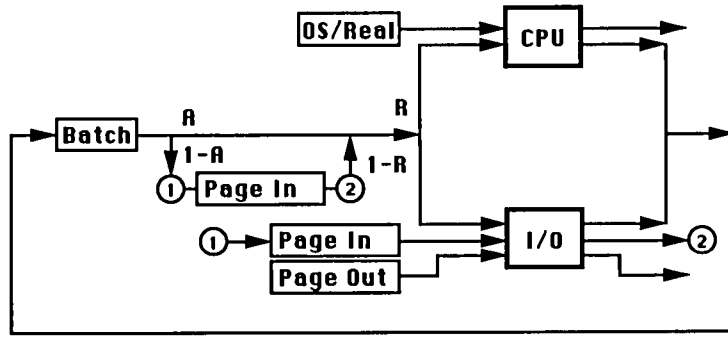
図4　走行ラン・モデル(C型モデル)

イ．バッチランはCPUとI/O装置を交互に使用する．また，CPUもI/O装置も使用しない（スワップアウトされている）状態がある．スワップアウトされている状態とされていない状態の比率は1—A：A(インコア比率，0<A<1）とする．

ロ．OSはCPUだけを使用し，その優先度はバッチランよりも高い．

ハ．スワップアウト/インはI/O装置だけを使用し，その優先度はバッチランよりも高い．

【M型モデル】(図5)



A：インコア比率　　R：CPU比率

図5　走行ラン・モデル(M型モデル)

イ．バッチランはCPUとI/O装置を交互に使用する．また，CPUもI/O装置も使用しない（ページアウトされている）状態がある．ページアウトされている状態とされていない状態の比率は1—A：A(インコア比率，0<A<1）とする．

ロ．OSはCPUだけを使用し，その優先度はバッチランよりも高い．

ハ．ページアウト/インはI/O装置だけを使用し，その優先度はバッチランよりも高い．

CPU，I/O装置の1回当り使用時間，OSのCPU利用率，インコア比率などがモデルのパラメータになるが，これらは対象計算機のハードウェア性能，実測結果をもとに決定する．

4) シミュレーション結果

2200/600計算機を対象としたシミュレーション結果を図6に示す．CPU台数が2台，メモリ容量が16M語の構成において，同時走行ラン数が6とおり(1,5,10,20,40,60ラン)，CPU比率が8とおり (0.01,0.1,0.2,0.3,0.4,0.5,0.75,0.99) で，48とおりのシミュレーションを行った．図6(a)はCPU要求1回当り待ち時間，(b)はI/O要求1回当り待ち時間，(c)はCPU利用率の推定値である．メモリ待ち時間はバッチランのCPU対I/O比率によりCPUおよびI/O待ち時間に含めてある．計算の便宜のため，待ち時間をSUP値1分を使うための所要時間(SUP値1分当り所要時間)に変換した．(d)はCPU・SUP値1分当り所要時間，(e)はI/O・SUP値1分当り所要時間，(f)はTOTAL・SUP値1分当り所要時間である．

5) 待ち時間の計算

図6より，システム内の同時走行ラン数とCPU比率がわかれば，待ち時間またはSUP値1分当りの所要時間を求めることができる．しかし，図6では，同時走行ラン数とCPU比率の48の組み合わせしか求めていないので，中間の値は以下のいずれかの方法で計算する．

イ．中間の値は補間法で求める．

ロ．待ち時間(所要時間)を同時走行ラン数とCPU比率の関数として，曲線近似による定式化を行う．

定式化の場合(ロ.)は予測のための計算時間は短くてすむが，適当な近似関数が見つからない時は誤差が大きくなる．補間法(イ.)は高次関数で補間すれば誤差が少ないが，計算時間が多くかかる．したがって，どちらの方法で行うかは予測の時に使う計算機の性能と必要な予測精度により決定する．

3.4 所要時間の計算方法

1) 所要時間計算の考え方

SUP値予測モデルおよび待ち時間予測モデルから，各ランの使用SUP値とSUP値1分当りの所要時間を計算することができる．これから各ランの所要時間を計算することを考える．バッチランの所要時間は各ランの走行順序（スケジュール）により異なるので，スケジュールを考慮しながら所要時間を求める必要がある．

バッチランA，B，Cがあり，それぞれの開始時刻をt_{A1}，t_{B1}，t_{C1}とする．待ち時間なしで走行した（すなわち，所要時間＝TOTAL・SUP値）場合の終了時刻をt_{A2}，t_{B2}，t_{C2}とする（図7）．

図7は，待ち時間を考慮しない時の走行スケジュールであるが，区間L_2，L_3，

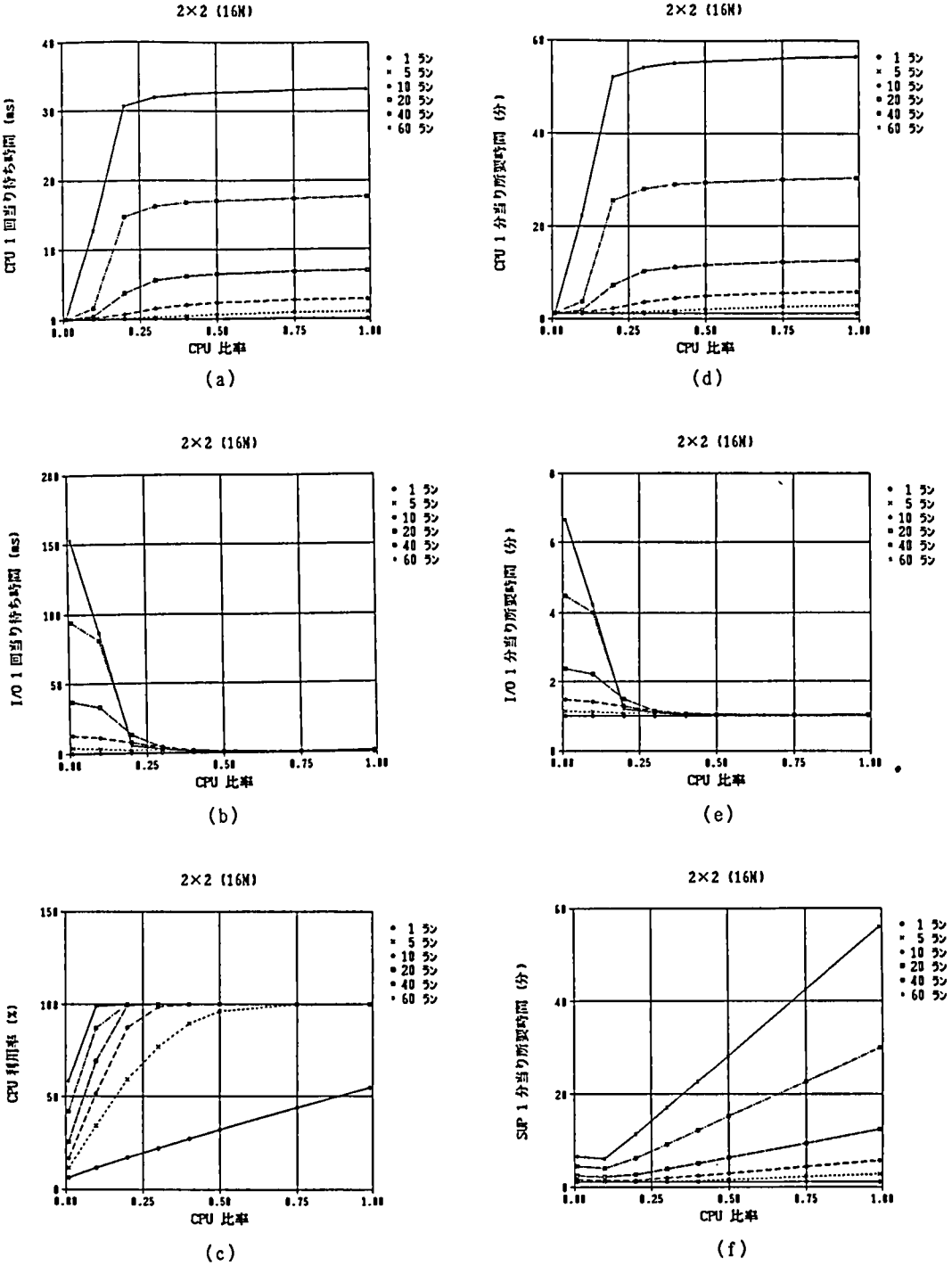


図 6　シミュレーション結果

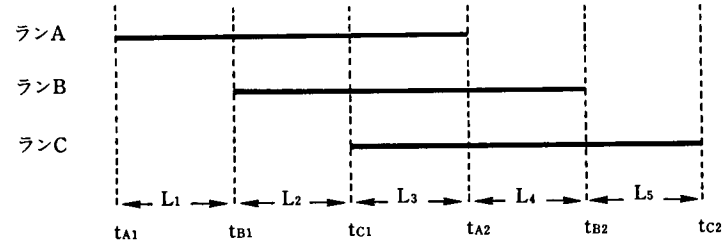


図 7　バッチラン A，B，C の走行スケジュール

L_4において複数のランが走行しているので，待ちが発生し各ランの所要時間は長くなる．各区間での同時走行ラン数と CPU 比率は計算できるので，待ち時間予測モデルより各ランの所要時間を求めることができる．求められた所要時間で再度，図 7 のようなスケジュールを作成すると，複数ランが走行している区間 L_2，L_3，L_4の長さが変わる．そこで，再度，所要時間を計算し，スケジュールを作成し直す．この計算を何度か繰り返し行い，最終的な所要時間を求める．実際には 5 回程度の反復で所要時間は収束する．

2)　所要時間計算アルゴリズム

所要時間計算のアルゴリズム（基本アルゴリズム）を以下に示す．このときバッチランのスケジュール（各ランの開始時刻と終了時刻）も同時に作成される．

【基本アルゴリズム】

① 各ランの SUP 値を予測する．SUP 値予測モデルにより，各バッチランの CPU・SUP 値，I/O・SUP 値，TOTAL・SUP 値を予測する．

② 初期スケジュールを作成する．TOTAL・SUP 値を所要時間として，ランネットワーク（各バッチランの走行順序）に従いバッチランのスケジュールを作成する．バッチランのスケジュール作成には PERT (Program Evaluation and Review Technique) の手法を用いる．

③ 所要時間を再計算する．

イ．区間別に CPU・SUP 値 1 分当りの所要時間を計算する．同時走行ラン数が変わる区間ごとに，待ち時間予測モデルを用いて CPU・SUP 値 1 分当り所要時間を計算する．

ロ．ラン別の CPU・SUP 値 1 分当りの所要時間を計算する．イ．で求めた区間別 CPU・SUP 値 1 分当り所要時間を区間の長さを重みとして加重平均し，各ランの CPU・SUP 値 1 分当り所要時間とする．

ハ．CPU を使用するための所要時間を求める．ロ．で求めたラン別の CPU・SUP 値 1 分当り所要時間と①の CPU・SUP 値を掛けて CPU を使用するための所要時間を求める．

ニ．I/O を使用するための所要時間を求める．イ．からハ．と同様の方法で求める．

ホ．ハ．とニ．の所要時間を加えてバッチランの所要時間とする．

④　③の所要時間とランネットワークをもとに，PERT 手法によりスケジュールを作成し直す．

⑤　③と④を全ランの所要時間が変化しなくなるまで繰り返す．

A，B，Cの3ランが走行する図7を例として，基本アルゴリズムの計算方法を図8に示す．

```
① ＳＵＰ値を予測する。予測したＳＵＰ値を以下のようにする。
                              ランA      ランB      ランC
                              ＝＝＝      ＝＝＝      ＝＝＝
    ＣＰＵ・ＳＵＰ値          ＳＣA      ＳＣB      ＳＣC
    Ｉ／Ｏ・ＳＵＰ値          ＳＩA      ＳＩB      ＳＩC
    ＴＯＴＡＬ・ＳＵＰ値      ＳＴA      ＳＴB      ＳＴC
② ＳＴA、ＳＴB、ＳＴCを所要時間としてバッチランのスケジュールを作成する。
③ 所要時間を再計算する。
  イ. 区間別にＣＰＵ・ＳＵＰ値１分当り所要時間を計算する。
      区間１　ＵＣ1＝ｆ（ｎ1，Ｃ1）
      区間２　ＵＣ2＝ｆ（ｎ2，Ｃ2）
      区間３　ＵＣ3＝ｆ（ｎ3，Ｃ3）
      区間４　ＵＣ4＝ｆ（ｎ4，Ｃ4）
      区間５　ＵＣ5＝ｆ（ｎ5，Ｃ5）
          ただし、ＵＣi：区間ｉのＣＰＵ・ＳＵＰ値１分当り所要時間
                  ｆ　：待ち時間予測モデル
                  ｎi：区間ｉの同時走行ラン数
                  Ｃi：区間ｉのＣＰＵ比率
                      Ｃ1＝ＳＣA／ＳＴA
                      Ｃ2＝（ＳＣA／ＳＴA＋ＳＣB／ＳＴB）／２
                        ：
  ロ. ラン別にＣＰＵ・ＳＵＰ値１分当り所要時間を計算する。
      ランA　ＵＣA＝（Ｌ1ＵＣ1＋Ｌ2ＵＣ2＋Ｌ3ＵＣ3）／（Ｌ1＋Ｌ2＋Ｌ3）
      ランB　ＵＣB＝（Ｌ2ＵＣ2＋Ｌ3ＵＣ3＋Ｌ4ＵＣ4）／（Ｌ2＋Ｌ3＋Ｌ4）
      ランC　ＵＣC＝（Ｌ3ＵＣ3＋Ｌ4ＵＣ4＋Ｌ5ＵＣ5）／（Ｌ3＋Ｌ4＋Ｌ5）
          ただし、ＵＣj：ランｊのＣＰＵ・ＳＵＰ値１分当り所要時間
  ハ. ラン別にＣＰＵを使用するための所要時間を求める
      ランA　ＥＣA＝ＵＣA×ＳＣA
      ランB　ＥＣB＝ＵＣB×ＳＣB
      ランC　ＥＣC＝ＵＣC×ＳＣC
          ただし、ＥＣj：ランｊがＣＰＵをＳＣjだけ使用するための
                         所要時間
  ニ. イ.からハ.のと同様の方法で、ラン別にＩ／０を使用するための所要時間を求める。
      ランA　ＥＩA＝ＵＩA×ＳＩA
      ランB　ＥＩB＝ＵＩB×ＳＩB
      ランC　ＥＩC＝ＵＩC×ＳＩC
          ただし、ＥＩj：ランｊがＩ／ＯをＳＩjだけ使用するための
                         所要時間
  ホ. 各ランの所要時間を求める。
      ランA　ＥA＝ＥＣA＋ＥＩA
      ランB　ＥB＝ＥＣB＋ＥＩB
      ランC　ＥC＝ＥＣC＋ＥＩC
          ただし、　Ｅj：ランｊの所要時間
④ ＥA、ＥB、ＥCを所要時間としてスケジュールを作成し直す。
⑤ ③と④を全ランの所要時間が変化しなくなるまで繰り返す。
```

図 8　基本アルゴリズムの計算法

3)　所要時間の予測例

上記モデルで，ある日のバッチラン約 800 の所要時間を予測した結果を図9と図10に示す．図9は予測値対実績値の走行時間帯別相関図である．斜めの線は予測値と実績値が一致しているところを表している．相関係数は 0.67 ～ 0.99 と

高い値を示している．予測値よりも実績値がかなり大きくなっているランはデータの到着待ちが主な原因である．逆に，実績値に対し予測値が大きくはずれているランは，実績の所要時間よりもSUP値の方が大きくなっているランである．SUP値が実際の計算機使用時間ではなく計算上の値であるため，大量のI/Oを

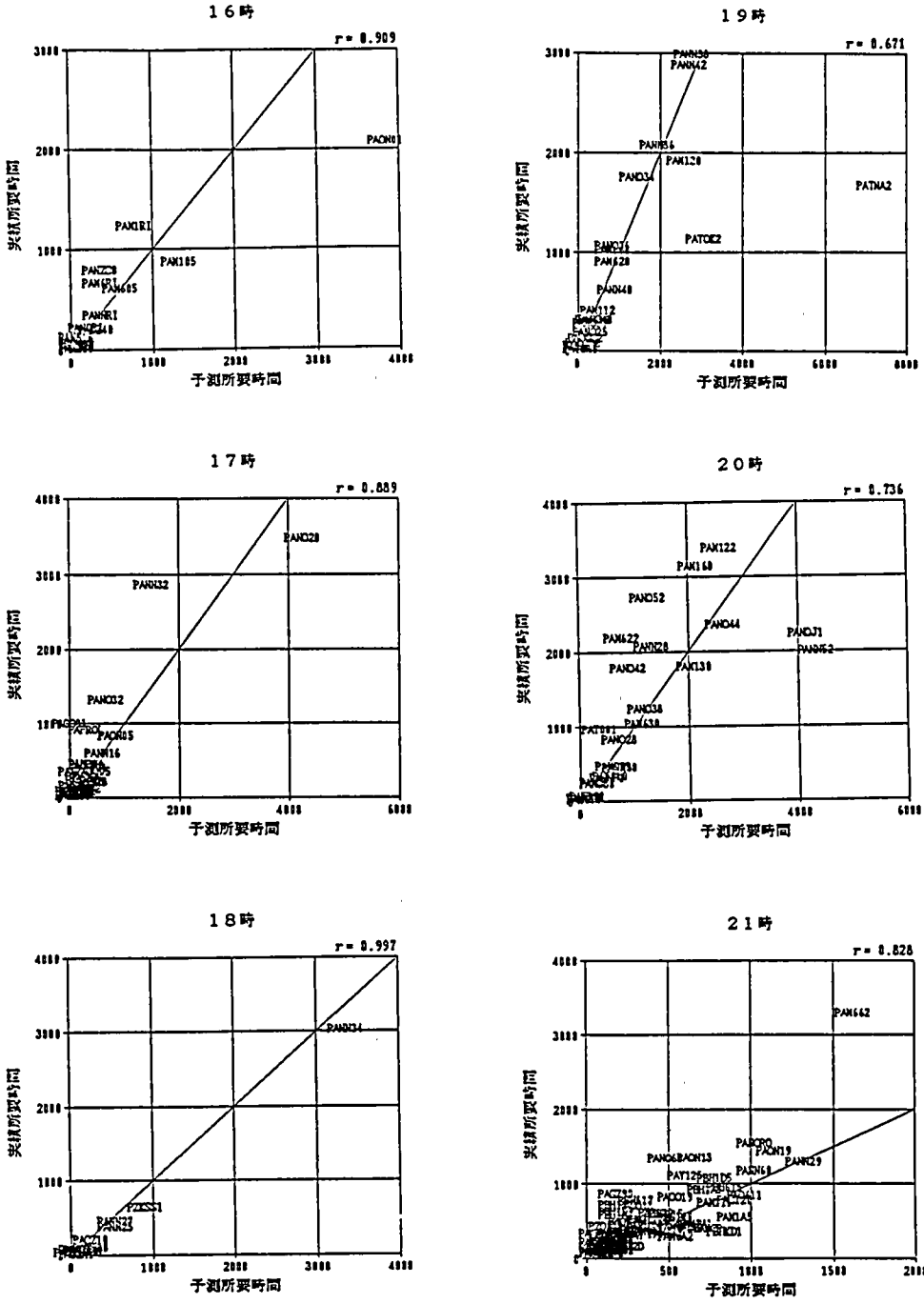


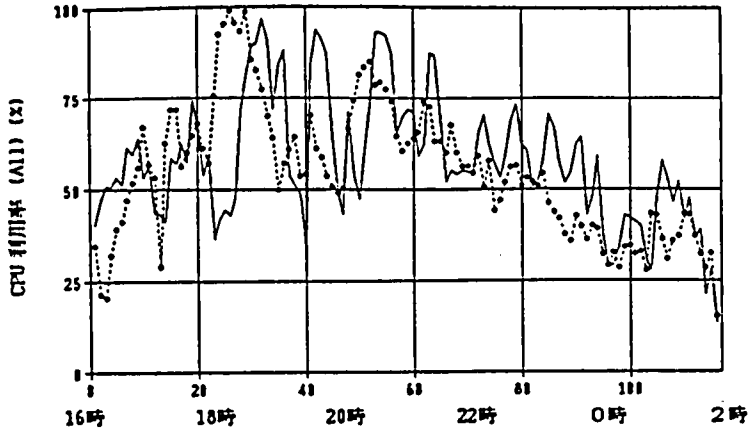
図 9　所要時間の予測結果

行うランなどは両者のずれが大きくなっているのが原因である．これらのランの予測精度をあげるためには原因別に個別に対応していく必要がある．しかし，全体的に予測値と実績値はほぼ一致しており，十分実用になることがわかる．

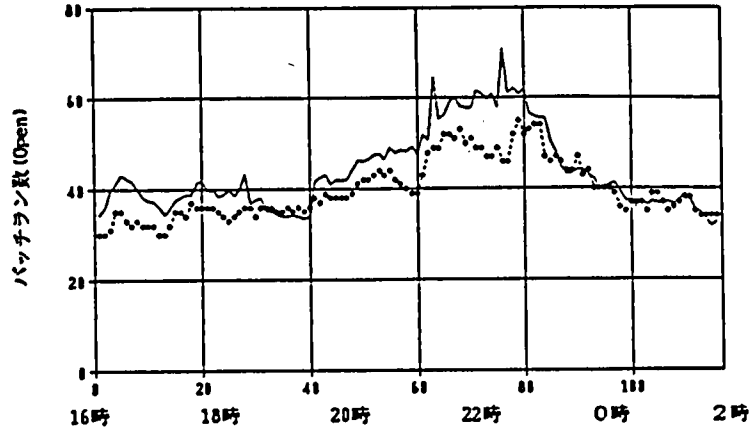
図10はCPU利用率(a)と同時走行ラン数(b)の時系列推移である．実線が実績値で点線が予測値である．これも予測値と実績値がほぼ一致しており十分実用になることがわかる．同時走行ラン数は予測値が実績値よりも全体に少なくなっているが，本モデルでの予測対象バッチラン以外にファイル転送のための短いランが1500ランほど走行しているためである．また，CPU利用率のピークと同時走行ラン数のピークが異なる時間帯にあり，同時走行ラン数だけでは予測できないことがわかる．

4)　所要時間予測における問題点

所要時間の予測において問題になるのは，バッチランの所要時間がTOTAL・SUP値よりも短い場合がたびたび発生することである．OSのI/O・SUP値の計



(a)　CPU利用率



(b)　同時走行ラン数

図10　CPU利用率，同時走行ラン数の予測結果

算式が実際のI/O時間と乖離していることや，CC/ER・SUP値が信頼のおけないものになっていることが主な原因である．このような場合には，所要時間の予測値をもとにSUP値を補正することが必要である．

3.5 デッドライン運転時の所要時間予測

1) デッドラインバッチの優先度

通常のバッチ運転時に，デッドライン（ランの終了期限）の付いたバッチラン(デッドラインバッチ)が1ラン走行している場合の所要時間の予測について考える．デッドラインバッチは終了時刻を早めるため，ランの優先度を一般のバッチランよりも高くしたものである．デッドラインバッチの優先度を以下のように仮定する．

イ．CPU使用の優先度は一般のバッチランよりも高く，OS/リアルタイムランよりも低い．

ロ．I/O使用の優先度は一般のバッチランと同じ．

ハ．メモリ使用の優先度は一般のバッチランよりも高い．

I/Oの優先度を一般のバッチランよりも高くしているシステムもあるが，ここでは，一般のバッチランと同じ場合を取り扱う．

2) デッドライン運転走行モデル

バッチランは以下のような構成で走行しているものとする．

	ラン数	CPU比率
イ．一般のバッチラン	n	Cb
ロ．デッドラインバッチ全体	$\frac{1}{n+1}$	$C=\frac{Cd}{(nCb+Cd)/(n+1)}$

3) 所要時間予測モデル

以上の仮定において，CPU待ち時間は，デッドラインバッチはほぼゼロとなり，一般のバッチランは増大する．I/O待ち時間は優先度が同じなので両者とも同じである．CPU利用率は処理の順序が変わるだけなので，通常運転時と同じである．したがって，各待ち時間およびCPU利用率は以下のように計算できる．

① 一般のバッチランの待ち時間

イ．CPU待ち時間は，n, Cb, Cdの関数になるので，シミュレーションによりこの関係を求める．

ロ．I/O待ち時間は，同時走行ラン数がn+1，CPU比率がCの時の通常運転時と同じになる．

② デッドラインバッチの待ち時間

イ．CPU待ち時間は，0となる．

ロ．I/O待ち時間は，同時走行ラン数がn+1，CPU比率がCの時の通常運転時と同じになる．

③ CPU利用率

CPU利用率は，同時走行ラン数がn+1，CPU比率がCの時の通常運転時と同じになる．

4. バッチランのスケジュールモデル

4.1 バッチランのスケジュール作成方針

バッチランのスケジュール作成において重要なことは，個々のランの所要時間（ターン・アラウンド)を短くすることではなく，全体の終了時刻を早くすることである．そのためには，単位時間当りの計算機の処理量（スループット）を上げる必要がある．

スループットを悪くする原因の一つは，OSの過負荷である．同時走行ラン数が多くなると，スワップアウトまたはページアウトが発生し，OSが過負荷になりスループットは悪くなる．もう一つの原因は，計算機の稼働率が低く，能力を最大限に利用していない場合である．同時走行ラン数が少ないと計算機は稼働率が低くなるので，OSが過負荷にならない程度にラン数を増やす必要がある．使用する装置に片寄りがある時にもスループットは悪くなる．バッチランがCPUとI/Oの二つの装置を使う場合，どちらかの装置に使用頻度の片寄りが生じた場合，片方の装置での待ちが多くなり，もう片方の装置が遊んでしまうことになる．バッチラン全体のCPU使用量とI/O使用量は一定であるので，システム内のCPU比率を一定（平均値）に保って運転することにより回避することができる．

以上より，バッチラン全体の終了時刻を早くするために，以下の方針でスケジュールを作成することにする．

イ．バッチランはスワップアウトまたはページアウトが発生しないラン数までは，できるだけ多く同時に走行させる．

ロ．システム内のCPU比率の管理限界を定め，管理限界内で運転するようにする．

同時走行ラン数の上限やCPU比率の管理限界は，実計算機での測定結果やシミュレーションにより決定する．

4.2 スケジュール作成の方法

1)　通常運転時のスケジュール

スケジュール作成時の同時走行ラン数やCPU比率の制御は，PERTの山くずし手法を応用して行う．まず，3.4項2)の基本アルゴリズムによりバッチランのスケジュールを作成する．次に同時走行ラン数またはCPU比率が管理限界内にないランをそのランの余裕時間（最早開始時刻から最遅開始時刻の間）内で，全体の終了時刻が最も早くなる時刻に開始時刻を移動する．これを全ランについて繰り返し，全てのランを管理限界内におくことによりスケジュールを作成する．

すなわち，以下の方法（山くずしアルゴリズム）でスケジュールを作成する．

【山くずしアルゴリズム】

① 基本アルゴリズムによりバッチランのスケジュールを作成する．

② 同時走行ラン数の山くずしを行う．

イ．同時走行ラン数が上限数以上になっている時間帯において，その時間帯に走行しているすべてのランの開始時刻を余裕時間内で移動して，全体の終了時刻が最も早くなるランと開始時刻を探す．

ロ．イ.で見つけたランをその開始時刻に移動する．

ハ．全体の終了時刻を早くするランがなくなるまでイ.，ロ.を繰り返す．

③ ②と同様の方法でCPU比率の山くずしを行う．

④ 全体の終了時刻を早くすることができなくなるまで②，③を繰り返す．

この方法は，②，③の山くずしの各ランの開始時刻の移動の過程において，基本アルゴリズムによる収束計算を行わねばならないので，膨大な計算量を必要とする．実際のスケジュールシステムでは，計算回数を減らす工夫が必要である．

2) デッドライン運転時のスケジュール

終了時刻を早くするためにデッドラインを導入する方法がある．特定の業務や特定のランの終了時刻を早くするためにもこの方法は有効である．バッチラン全体の終了時刻を早めるためには，クリティカルパス上のラン（余裕時間が0のラン）にデッドラインを付ければ良い．ただし，あるランにデッドラインを付けて処理時間を短くしても，他のランの処理時間が長くなるので，全体の終了時刻が早くなるとは限らない．また，あるランにデッドラインを付けるとクリティカルパスが変わる場合がある．これらの点を考慮して，以下のような手順（デッドライン・アルゴリズム）でデッドラインを導入する．

【デッドライン・アルゴリズム】

① 基本アルゴリズムによりスケジュールを作成する．

② クリティカルパス上の先頭のランから試してみて，全体の終了時刻を早くする最初の1ランにデッドラインを付ける．

③ デッドラインを付けたラン以降を基本アルゴリズムでスケジュールし直す．

④ スケジュールし直したランに対し，②の方法で一つのランにデッドラインを付ける．

⑤ デッドラインを付けるランがなくなるか，最後のランにデッドラインが付くまで③，④を繰り返す．

山くずしアルゴリズムとデッドライン・アルゴリズムを組み合わせることにより，最適なスケジュールを作成することできる．一般に，各ランの走行順序や走行時刻がランネットワークにより厳しく規定され走行順序の自由度が小さい場合は，デットライン・アルゴリズムの方が有効である．一方，各ランの走行順序や走行時刻が比較的自由に設定できる場合は，山くずしアルゴリズムが有効になる．

4.3 スケジュールの作成例

ある日に実際に走行した約600のバッチランに対して各アルゴリズムでスケジュールを作成し，その結果を比較した．各スケジュールの所要時間（最初のランを開始してから最後のランが終了するまでの時間）を表1に示す．ケース0は基本アルゴリズムによるスケジュールである．ただし，各ランネットワークの先頭ランの開始時刻を実際の開始時刻に合わせている．所要時間は357分であるが，当日の実績は348分でほぼ同様の結果が得られた．したがって，これを実績とみなして各スケジュールがどれだけ早くなったかを検討する．なお，実際の開始時刻を考慮しないで基本アルゴリズムを適用した場合の所要時間は361分である．

ケース1は同時走行ラン数だけに上限(20ランと30ラン)を設定し，山くずしアルゴリズムを適用した結果である．ランネットワークの制約が厳しくラン走行順序の自由度が少ないため，ケース0の実績にくらべそれほど改善されていない．ケース2は

表1　各スケジュールにおける所要時間

ラン数上限	ケース0	ケース1	ケース2	ケース3
20ラン	357分	352分	337分	293分
30ラン	357分	354分	329分	301分

同時走行ラン数の上限設定に加え，CPU比率にも管理限界を設定した場合である．6～7%程度の改善がなされている．ケース3はデッドライン・バッチを導入したスケジュールである．16～18%程度改善されている．すなわち，約6時間かかる処理が約5時間で終わることを表しており，このケースではデッドライン導入の効果が大きいことがわかる

5　複数計算機のスケジュール

4章は単一計算機におけるスケジュール作成の方法である．通常，運用管理システムを必要とするような大規模システムの場合，複数の計算機を連携して運用している場合が多い．以下に，複数計算機におけるスケジュール作成の方法について記述する．

5.1　複数計算機の連携

図11はA，Bの2台の計算機のランネットワークである．2台の計算機はランB2からランA3，ランA5からランB5にデータを受け渡すことで連携している．すなわち，ランA3，B5はランB2，A5の後続ランとなっている．したがって，バッチランのスケジュールを作成する場合には，相手の計算機のスケジュールも考慮する必要がある．データの受け渡しは以下のいずれかの方法で行われるものとする．

イ．テープを媒体として受け渡す．

ロ．チャネルまたは通信回線経由のファイル転送で受け渡す．

ここで，データの受け渡しも一つのバッチランとみなせば，複数の計算機のランネットワークを一つに統合することができる．統合したランネットワークのスケジュールを作成すれば全計算機の連携を考慮したスケジュールを作成することできる．

以下に，複数計算機システムにおける所要時間予測モデルとスケジュール作成の方法について述べる．

5.2　複数計算機の所要時間予測モデル

複数計算機のスケジュールを作成するためには，各バッチランの所要時間とデータ連携の所要時間を予測する必要がある．各所要時間の予測モデルは以下のように作成する．

1)　バッチランの所要時間予測モデル

バッチランの所要時間予測モデルは，単一計算機の場合とまったく同じモデルを作成する．すなわち，各計算機ごとにSUP値予測モデルと待ち時間予測モデルを独立に作成する．

2)　データ連携の所要時間予測モデル

データ連携の所要時間はデータの受け渡し方法により異なる．

イ．テープを媒体として受け渡す場合

テープを媒体として受け渡す場合は，テープの移動のための一定時間を考

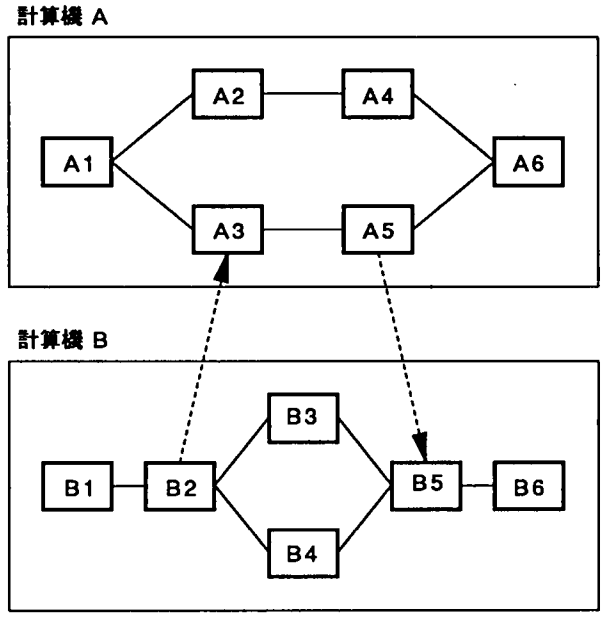


図 11　複数計算機の連携

えれば良い．また，遠隔値からの移動など，移動のためのスケジュールが決められている場合は，一定の時刻を考えれば良い．

ロ．ファイル転送による場合

ファイル転送による場合は，転送データ量により処理時間が変動する．転送データ量も SUP 値と同様に月，日，曜日等により変動していると考えられる．したがって，ファイル転送の所要時間は，SUP 値の予測と同様の統計モデルにより予測を行うことができる．

5.3　複数計算機のスケジュール作成方法

データの受け渡しも一つのバッチランとみなし，全計算機のランネットワークを一つに統合する．この統合したランネットワークを対象に，全体を一つの計算機と考えて，スケジュールを作成する．

すなわち，以下の方法（複数計算機アルゴリズム）でスケジュールを作成する．

【複数計算機アルゴリズム】

① 複数計算機それぞれのランネットワークを一つのランネットワークに統合する．

② 全体を一つの計算機とみなして，山くずしアルゴリズムにより全バッチランのスケジュールを作成する．このとき，待ち時間予測モデルはそのランが走行する計算機のモデルを使用する．

③ 全体のクリティカルパスに対して，デッドライン・アルゴリズムによりデッドラインを導入する．

④ 個々の計算機それぞれのクリティカルパスに対して，デッドライン・アルゴリズムによりデッドラインを導入する．

⑤　全体の終了時刻を早くすることができなくなるまで③，④を繰り返す．

5.4　複数計算機への適用例

4台の計算機が連携してバッチ処理を行っているシステムがある．このシステムの過去9日間のバッチ処理の終了時刻は表2のとおりであった．

表2　実績の終了時刻

号機	1日 (月)	2日 (火)	3日 (水)	4日 (木)	5日 (金)	6日 (土)	7日 (日)	8日 (月)	9日 (火)
1号機	6：34	5：07	6：39	5：32	4：31	0：31	0：32	5：56	5：35
2号機	6：31	4：39	6：08	5：31	5：33	0：44	0：47	5：00	6：11
3号機	6：32	5：09	6：12	4：46	4：46	0：25	0：24	5：14	4：44
4号機	6：32	4：17	6：12	3：59	3：59	0：23	0：22	4：16	4：20

このシステムで10日目（水曜日）に走行する 2,722 ランについて終了時刻の予測とデッドライン導入によるスケジュールの作成を行った．さらに，作成したスケジュールでデッドラインを付けられたバッチランに実際にデッドラインを付けて走行させた．予測と実際の終了時刻をを表3に示す．

表3　10日の予測/実績終了時刻

		(A)	(B)	(C)	
号機	ラン数	予測 終了時刻	デッドライン導入 予測終了時刻	デッドライン導入 実績終了時刻	デッドライン バッチラン数
1号機	1,061	5：54	5：44	5：35	14
2号機	912	5：13	4：37	4：49	27
3号機	392	4：20	3：42	4：13	16
4号機	357	4：04	3：29	3：51	15

表3の予測終了時刻(A)はデッドラインを付けないで運用した場合の終了時刻の予測値である．(B)はデッドラインを付けて運用した場合の終了時刻の予測値で，(C)は実績値である．

デッドライン導入による予測短縮時間は，それぞれの計算機で10分，36分，38分，35分となっている．1号機はデッドライン導入の効果が小さいが，1号機は最近更新したばかりの新しい計算機で，能力に十分余裕があるためである．

実績の終了時刻は，どの号機も通常運転時の予測結果よりも早くなっている．また，土曜日と日曜日を除けば，過去9日間の実績に比べて早く終了している．以上より，バッチ処理の効率的運用のためには，デッドラインの適切な導入が効果的であることがわかる．

今回の例は，すでに厳しいランネットワークが定義されている場合に対して適用したものであり，スケジューリング余裕度の低いケースであったと考えられる．今後ランの開始時刻等に余裕度があるケースに適用して，その効果を計ることが課題である．

6. おわりに

バッチ処理を効率的に運用しようとする背景には，計算機資源の有効活用を計り，計算機システムに対する過剰な投資を抑えようとすることがある．バッチランの走行

スケジュールを適切に作成し，処理全体の終了時刻を早めることができれば，計算機の規模や更新の時期の適切な見積りが可能となる．また，日々の運用においては，処理時間短縮による費用の削減，バッチ業務の遅れによる翌日業務への影響の事前回避などが可能となる．

本稿の検討で使用したプログラムは，UNIX*ワークステーションUSファミリ上で稼働している．ホスト計算機とは回線で接続されており，定期的にバッチランの走行状況を貰いながら，スケジュールの予測や作成を行うようになっている．補助的な機能として，ランネットワークの表示や各バッチランの詳細情報の表示が行えるようになっている．しかし，作成したスケジュールどおりにバッチランを走行させるためには，運用管理システムとの連携が必要であり，今後の課題である．

---

* UNIXオペレーティングシステムは，UNIX System Laboratories, Inc. が開発し，ライセンスしている．

執筆者紹介　松　田　芳　雄 (Yoshio Matsuda)

1974年慶応義塾大学工学部管理工学科卒業．同年日本ユニシス(株)入社．オペレーションズ・リサーチ，統計解析関係のシステム開発に従事．現在研究開発室に所属．日本オペレーションズ・リサーチ学会，日本シミュレーション学会，情報処理学会会員．

河　岸　憲　一 (Ken-ichi Kawagishi)

1971年京都大学工学部卒業．同年日本ユニシス(株)入社．以来燃焼管理等の原子力アプリケーションを始めとする電力関連業務に従事．現在 中部支社 社会公共システム部に所属．

ニューラルネットワークにおける学習の高速化と収束安定性の考察

Some Thoughts over High-speed Learning and Convergence Stability on the Neural Network

浦 上 浩 一

要 約 ニューラルネットワークの学習アルゴリズムとして提案された誤差逆伝播（Back Propagation，以下BPと略す）学習アルゴリズムについて，学習速度の高速化，局所最小値への落込みの解決法がいくつか提案されている．その中でGA学習法，最適初期値設定法，局所最小値回避法，高速BP学習法を用いて実験を行った．学習問題としてXOR問題とノイズを含む株価時系列学習問題を取り上げ検証を行った．

最適初期値設定法に関しては両方の問題で効果が見られた．XOR問題で顕著な効果のあったGA学習法はノイズを多く含んでいると思われる株価問題では効果が見られなかった．またどの手法においても，試行錯誤によりパラメータの最適な値を見つけなければならなかった．

今回は収束速度，局所最小値回避を評価基準として取上げたが，今後汎化能力の評価も行ってみたい．

Abstract Several methods have been proposed to solve problems pertaining to learning speeds, and to avert false local minimums for the back propagation (BP) learning algorithm that serves for the neural network. The author conducted experiments with four methods, among them, such as (1) genetic algorithm (GA) learning, (2) the setting of optimized initial synapse values, (3) the avoiding of false local minima, and (4) high-speed BP learning for evaluation of each solution method with the use of an XOR problem and a noise-included time-series stock price problem as chosen exercises.

The method of setting optimized initial synapse values has proved to be effective for both problems, while GA learning known as being remarkably helpful for the XOR problem has turned out to be of almost no effect on the stock price problem supposedly possessed of a gread deal of noisy data. In addition, each method required the author to determine optimized parameter values on a trial-and-error basis.

This paper is specifically focused on the speed of convergence and the avoiding of false local minima as evaluation criteria. The author also intends to make a further evaluation in the near future in terms of the algorithm's generalization capability.

1. はじめに

ニューラルネットワークの学習アルゴリズムとして提案されたBP学習アルゴリズムによる情報処理の研究は近年盛んに行われ，複雑なパターン分類問題に対する有効性が確認されている．しかし，依然として学習速度，局所最小値への落ち込みの問題等が指摘されており，これらに対してさまざまな改善案が報告されている．ところがこれらの報告を見るとXOR問題，パリティ問題等での実験に留まり，ノイズ(不規則

変動や計測値誤差）を含んだような実問題での検証が行われているものは少なく，またいくつかの手法を組み合わせた効果に関して論じたものはほとんどない．

本稿では，提案されているいくつかの方法を取り上げ，実問題での実験，および相乗効果の検討を行った．

ここでは次の4手法を取り上げた．

1) GA（遺伝的アルゴリズム）学習法……BPの代わりに初期収束が速く，収束安定性が高いといわれているGAを使う手法（文献[1]のニューラルネットワークへの適用）．

2) 最適初期値設定法……初期値の設定方法によって局所最小値への落ち込みを避ける手法[5]．

3) 局所最小値回避法……学習中にシナプス値を監視し，調整することによって局所最小値への落ち込みを避ける手法[3],[4]．

4) 高速BP学習法……従来の誤差評価関数（二乗誤差和）を変えることによって学習速度を向上させる手法[2]．

2. 各手法の概要

2.1 GA学習法

文献[1]で提案されている実数値最適化遺伝的アルゴリズム手法を，シナプスウエイトの学習へ適用した．初期収束の速さが特徴としてあげられる．

ニューラルネットワークのシナプスウエイトの組を一つの個体に対応させ，シナプスウエイトを個々の実数値を持つ遺伝子に直接対応させる（図1に対応の例を示す）．

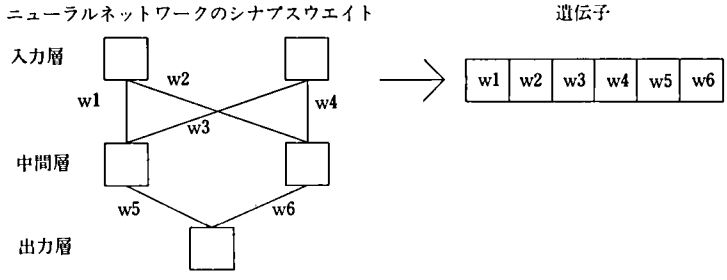


図1 シナプスウエイトと遺伝子の対応の例

次の操作を行い学習（最適化）を行う．

1) 個体の発生……実数値を指定された幅でランダムに設定しN（パラメータによって与えられる）個の個体を発生させる．

2) 最優良個体の次世代への保存……適合度の最も高い個体をそのまま次世代に残す．

3) クロスオーバー……適合度に比例する確率で二つの親を選択し，指定された比率にしたがって2種類のクロスオーバー（線形，二点）を行う．

線形クロスオーバーは次のように行い，生成された三つの子（G1', G2', G3'）のうち適合度の高い二つを選択する．

クロスオーバー前

$G1=[\ 0.4\quad 0.8\quad 3.2\quad -3.0]$

$G2=[-2.0\quad 3.0\quad -2.0\quad 6.0]$

クロスオーバー後

$G1'=1/2*G1+1/2*G2$

$=[-0.8\quad 1.9\quad 0.6\quad 1.5]$

$G2'=3/2*G1-1/2*G2$

$=[1.6\quad -0.3\quad 3.8\quad -7.5]$

$G3'=-1/2*G1+3/2*G2$

$=[-3.2\quad 4.9\quad -4.8\quad 10.5]$

x の適合度を f(X) とした場合，

仮に　$f(G1')=0.9$,

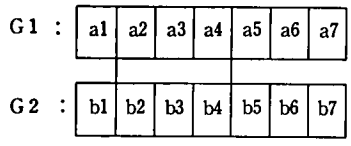
$f(G2')=0.5$,

$f(G3')=0.7$ とすると

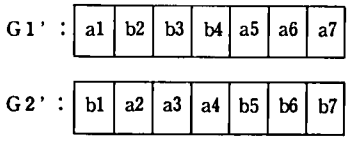
G1' と G3' が選択される．

二点クロスオーバーはランダムにクロスオーバーポイントを二点選択し，二つの親から二つの子を生成する．具体的には次のようになる．

クロスオーバー前



仮にクロスオーバーポイントを 1,4 とすると，クロスオーバー後



G2' となる．

4)　ミューテーション……ミューテーション率にしたがって，指定された幅（たとえば−5.0〜5.0）の範囲でランダムに調整値を決め，値を調整する．

5)　終了判定……最も良い適合度が基準値に達するか，最大繰り返し回数に達するまで 2) に戻って繰り返す．

2.2　最適初期値設定法

シナプスウェイトとしきい値の初期値は，従来はある範囲の乱数を与える方法が一般的で，収束しなかった場合はうまく収束するまで初期化し，やり直す方法が用いられている．また，この方法では入力数を考慮しないため，入力数によっては最初から飽和してしまう点，入力パターンによっては分離能力が低下してしまう等の問題点があった．このため，従来の初期値の決定法は学習速度の低下や局所最小値への落ち込みなどの原因になると考えられている．文献[2] により，提案されている最適初期値設定法の要点は次のとおりである．

xi をユニットの入力，wi をシナプスウエイト，$w0$ をしきい値，y をユニットの出力で0から1の値をとるものする．

入力 xi と出力 y を合わせた空間の $y=\alpha$ を満たす部分集合 $L(\alpha)$ は

$$L(\alpha):\ y=\Phi(w_0+\Sigma wixi)=\alpha$$

で表せ，$L(\alpha)$ の入力空間（$x1, x2, \cdots, x_n$）への射影 $Lp(\alpha)$ は，

$$Lp(\alpha):\ w0+wixi=\Phi(-1)(\alpha)$$

となる．

ここで次のようにシナプスウエイトとしきい値を設定する．

1) ニューロンユニットの出力の変化の大きい領域（活性領域）の幅がそのユニットの入力空間超キューブのサイズ（対角線の長さ）と整合するようにユニットごとにシナプスウエイトとしきい値を設定する．

2) 次に，$Lp(0.5)$が入力空間超キューブの中心を通るように，しきい値を調整する．

2.3 局所最小値回避法

文献[3],[4]によれば局所最小値回避法は次のとおりである．学習が収束した状態とは，シナプスウエイトの更新量が十分小さくなり，学習を続けてもネットワークのパフォーマンスが変化しなくなった状態であるといえる．したがって局所最小値への収束とは，シナプスウエイトの更新量が小さくなったにもかかわらず，特定の入力に対する誤差が許容値を上回る状態であると定義できる．

BPの定義式から考えると更新量が小さくなる要因は二つあり，一つは誤差が小さくなる場合，もう一つはユニットの出力が飽和してシグモイド微係数が小さくなる場合である．

当手法は学習中の一定期間ごとにシナプスウエイトの活性化を行い，誤差が大きい間は微係数をある程度の値以上に保つものである．活性化係数は出力層における最大誤差，学習係数，ユニット飽和度を考慮し設定する．

2.4 高速BP学習法

文献[5]によれば高速BP学習法は次のとおりである．出力関数がシグモイド関数の場合，学習の終盤になりユニットの出力値が1や0に近づくとシグモイド関数の導関数の値は0に近づき，シナプスの修正量は小さくなり学習が進まなくなる．

一般には評価関数は平均二乗誤差を用いるが，当手法ではシナプスウエイトの更新式でシグモイド関数の導関数を打ち消すような評価関数を用いることによって学習速度の低下を防止する．

教師信号が0,1に近い数値である場合に高速化の効果がある．

3. 学 習 問 題

次の二つの問題で各手法の効果を検討した．

3.1 XOR問題

入出力パターンは次のとおりである．

入力1	入力2		出力
1	1	⟶	0

1	0	⟶	1
0	1	⟶	1
0	0	⟶	0

誤差が既定値を下回った場合に，収束したと判断し学習を停止する．学習回数が20000回に達しても収束しなかった場合の学習回数は20000回として扱う．この試行を1000回行い，全試行の学習回数の平均とそのうちの収束したものの学習回数の平均をとる．

3.2 株価時系列の学習問題

テクニカル分析結果の8個のデータを入力とし，1日後の株価の上昇率を教師信号として与えている．つまり8入力1出力のニューラルネットワークで(詳細は文献[7])，入力，出力とも0〜1の実数値をとる．ただし学習の対象とした期間では株価の変動が激しくて教師信号が0ないし1に近い数値になることが多かった．

テクニカル分析は古くから株価の予測に用いられており，「ある分析値が××であれば株価が上昇する」というようなルールが唱えられている．しかし，このようなルールにしたがって株価が動く場合も，そうならない場合もあり，ノイズの多い問題としてこの問題を取上げた．

一回の試行の学習期間を25日，始点を1990年4月1日とし，始点を順に1日ずらして128回の試行を行う．誤差が既定値（エラーの二乗平均が0.01）を下回った場合に収束したと判断し学習を停止する．また学習回数が2000回に達しても収束しなかった場合の学習回数は2000回として扱う．128回の全試行の学習回数の平均とそのうちの収束したものの学習回数の平均をとる．

この株価の推移を図2に示す．

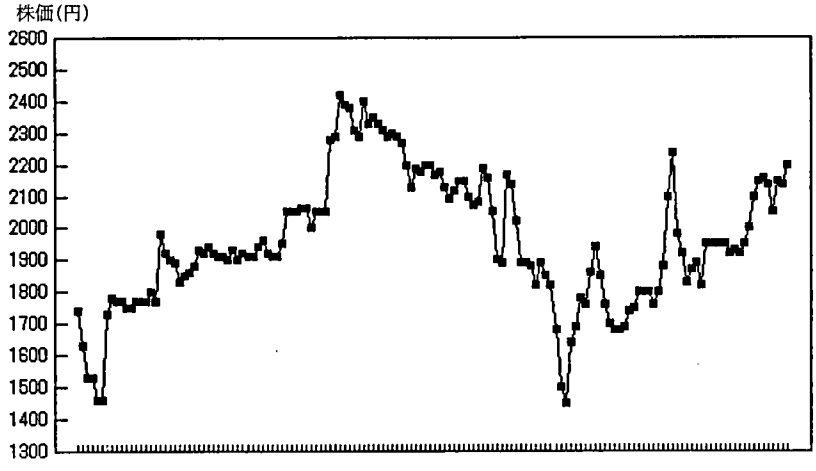


図 2　株価の推移（1990/4/1〜1990/10/31）

4. 各手法の組合せ効果の実験と評価

4.1 手法の組合せ

各手法の併用効果を検討するために，表1のように9種類の実験パターンを考える．

表1 実験パターン

実験パターン番号	採用手法			
	GA 学習法	最適初期値設定法	局所最小値回避法	高速 BP 学習法
0				
1	○			
2		○		
3			○	
4				○
5		○	○	
6		○		○
7			○	○
8		○	○	○

GA 学習法については他の手法との組合せが不可能なため，組合わせは考えず，実験パターン 1 のみで使用している．また実験パターン 0 は標準の BP でどの手法も用いていないことを表している．実験パターン 0,1 以外は標準 BP に採用手法を付加している．

4.2 評 価 方 法

GA 学習法とそれ以外の手法の学習速度は単純には比較できない．GA 学習法以外の手法の学習速度は学習回数の大小で比較できるため，試行したものの平均学習回数(平均学習回数)，収束したものの平均学習回数（収束平均学習回数）によって学習速度を評価した．また収束率（収束したものの回数/全試行回数）によって収束安定性を評価した．ただし，学習結果はシナプスウエイトの初期状態によって変ってくるため，多少の差(平均学習回数で 10%程度，収束平均学習回数で 5%程度，収束率で 1～2 ポイント程度）では効果があったとは評価できないと考えた．

4.3 問題ごとの実験結果

4.3.1 XOR 問題の結果

表 2 に実験結果を示した．実験パターン 1 は BP を用いていないため，学習速度については他の学習方式との単純に比較はできない．しかし 1000 回の試行すべてが収束しており，収束の安定性は最も良かった．

実験パターン 1 以外では，GA 学習法以外のすべての手法を用いた実験パターン 8 が最良の結果（平均学習回数，収束平均学習回数が最小で，収束回数が最大）となった．他の論文で示されている単独での効果とともに，他の手法との組合わせの効果も確認できた．

高速 BP 学習法に関しては実験パターン 4 の結果でもわかるように，収束平均学習回数は少なくなっているが，収束率が小さく収束安定性は低い．つまりシナプスウエイトの初期値によって結果が大きく変動するためである．これは最適初期値設定法との併用（実験パターン 6）で収束率が改善されていることからも判る．逆に高速 BP 学習法と局所最小値回避法の併用（実験パターン 7）は効果は見られず，局所最小値回避法のみを使用したもの（実験パターン 3）の方が学習速度も収束安定性も良い結果とな

表2　XOR 問題の実験結果

パターン	平均学習回数	収束平均学習回数	収束率
0	3983.4	909.9	83.9
1	101.5	101.5	100.0
2	2388.7	1635.7	95.9
3	1570.9	1000.9	97.0
4	4708.4	545.1	78.6
5	1915.1	1489.3	97.7
6	3382.0	654.3	85.9
7	2350.0	1041.9	93.1
8	539.1	441.3	99.5

っている．

最適初期値設定法は高速 BP 学習法と併用（実験パターン 6）することにより収束平均学習回数が改善され，局所最小値回避法と併用（実験パターン 5）することにより収束率が改善されている．

4.3.2　株価時系列の学習問題

表 3 に実験結果を示した．最適初期値設定法は単独で用いた場合（実験パターン 2）では標準 BP のみの場合（実験パターン 0）に比べて学習速度，収束安定性共に向上している．しかし他の手法との併用（実験パターン 5,6,8）効果は確認できない．

高速 BP 学習法は収束平均学習回数が少なくなり，収束率が低くなるといった傾向は XOR 問題と同じである．しかし教師信号が 0,1 の近辺の数値であったことから，収束平均学習回数の向上は XOR 問題ほどではなかった．

局所最小値回避法を単独で用いた場合（実験パターン 3）は実験パターン 0 と結果はほぼ同じで，効果は確認できなかった．これは学習データにノイズが多いために，もともと局所最小値に陥り難いことが原因として考えられる．

GA 学習法は収束率は 0 で XOR 問題のような収束安定性の効果はまったく見られなかった．原因としては，学習パターンの多さ（25 日分），ノイズが多いこと等があると考えられる．

表3　株価時系列学習問題の実験結果

パターン	平均学習回数	収束平均学習回数	収束率
0	1609.3	912.8	36.0
1	2000.0	——	0.0
2	1397.5	516.8	40.6
3	1601.1	913.5	36.7
4	1481.8	526.0	35.2
5	1388.0	493.5	40.6
6	1419.7	452.5	37.5
7	1541.8	456.7	29.7
8	1433.7	389.0	35.2

4.3.3 総 合 評 価

最適初期値設定法については，どちらの問題でも効果が確認できた．高速 BP 法は，教師信号が 0,1 (整数) に限られる場合には効果がある．しかし 0,1 以外の中間値も含まれる場合はその比率の増加にしたがって効果が薄れてくる．局所最小値回避法はノイズがない場合には効果があるが，ノイズが多い問題では効果が見られない．

以上から最適初期値設定法は常に用い，教師信号が 0,1 であることが多い場合には高速 BP 学習法を用い，さらに局所最小値に陥ることが多い場合には，局所最小値回避法を用いれば良いことが判る．

各手法のパラメータは問題によって変化させて，その手法が有効かどうかを判定しなければならなかった．また，単独で手法を使用した場合の最適なパラメータ値は，他手法を併用した場合の最適なパラメータと同一になるとは限らない．よって複数の手法を組合わせた場合は，最適なパラメータを見つけるのは時間のかかる作業であると考えられる．

5. お わ り に

株価問題では未学習のデータを入力し株価の予測に用いるために，汎化能力が重要である．したがって，複数の株価を取り上げ汎化能力の検討も必要である．またノイズの多い問題ではノイズを減少させる工夫（ノイズデータの排除，移動平均により不規則変動を除去するなど）も検討する必要がある．今後はこれらの課題を取り上げ研究を行うつもりである．

なお本稿は，通商産業省工業技術院 電子技術総合研究所 言語システム研究室における技術指導のもとに行った研究成果に基づいている．ご指導いただいた言語システム研究室の二木厚吉室長（現在 北陸先端科学技術大学院 大学教授）ならびに岡田技官に感謝の意を表する．

参考文献 [1] H. A. Wright, Genetic algorithms for real parameter optimization, In Foundations of genetic algorithms, (ed. Rawlins J. E.), pp. 205～218, Morgan Kaufmann, 1991.

[2] 吉川敏則, 河口陽子, バックプロパゲーション法を用いたニューラルネットワークの高速化学習アルゴリズム, 電子情報通信学会論文誌 92/4 Vol. J 75-D-2 No. 4 pp. 837-840.

[3] 福岡豊, 松木秀夫, 村岡史健他, 誤差逆伝搬学習則の収束成功率を高めるための一手法, 神経回路学会 第 3 回全国大会講演論文集 pp. 90～91.

[4] 福岡豊, 伊東輝顕, 南谷晴之他, False Local Minima を回避するための誤差逆伝搬, 神経回路学会 第 4 回全国大会講演論文集 pp. 120～121.

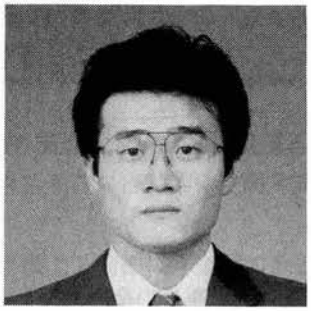
[5] 賈 棋, 戸田尚宏, 臼井支朗, ニューラルネットワークにおける逆伝搬学習アルゴリズムの初期値設定に関する一考察, 電子情報通信学会論文誌 D-2 Vol. J 73-D-2 No. 8 pp. 1179～1185.

[6] David E. Goldberg, : Genetic Algorithms in Search, Optimization, and Machine Learning. 1989.

[7] 浦上浩一, ニューラルネットワークの株式売買判断システムへの応用, 日本ユニシス技報 27 pp. 63～71.

執筆者紹介　浦　上　浩　一 (Koichi Urakami)

1984年九州大学経済学部 経済工学科卒業．同年日本ユニシス(株)入社．金融ユーザへのKEサービス，ニューラルネットワークを用いた株価予測システム開発などに従事．現在 システム技術第一本部 知識システム部応用システム課に所属．エキスパートシステム開発を担当．



知識ベースを利用した国際テレックス電文の自動解読

An Inter-bank Telex Message Handling System Using a Knowledge Base

中 田 純 一

要 約 海外の金融機関との資金決済や各種通信に使われる国際テレックス電文は，英語などの自然言語で書かれているが，その様式は統一されておらず，正確な解読には豊富な知識と経験をもった専門家が必要であった．

このたび㈱三和銀行において，この国際テレックス電文の解読処理を人工知能の手法を使って自動化した．テレックス電文解読システムの目的は，テストキー計算要素の抽出，および電文の表す取引種類の判別と配信先部署の判定である．

電文解読のためのアプローチとしては，システムに要求される出力情報が特定されていることに着目し，パターンマッチング手法を用いて必要な情報を個別に電文中から抽出するとともに，それらを組み合わせ最終的な出力結果を得るという方法をとった．そのために，テレックス電文固有の表現法や商習慣，および業務担当者のもつ知識やノウハウを業務領域固有の知識として知識ベース化した．知識ベースは，オブジェクトクラスとプロダクションルール，電文解析サブプログラム，キーワードファイル，辞書ファイルで表現，構成される．システムの性能評価のために，解読率という尺度を導入した．

本システムの特徴は，知識ベースを利用した一種の文書理解システムであること，および24時間稼働のリアルタイム型エキスパートシステムであることである．

Abstract Inter-bank telex messages used for all varieties of communications, including the settlement of accounts, with foreign financial institutions are usually written in natural language such as English, but so far have required human experts rich in knowledge and experience for accurate reading due to no uniformity of message formats.

Recently, the Sanwa Bank has succeeded in computerizing the reading and handling of inter-bank telex messages using artificial intelligence technology. The aims of the system are to extract necessary information for test-key calculation from messages, to identify the types of transactions indicated in messages and to automatically identiay appropriate departments they should be sent to.

Bolstered by the specifiable nature of the system's output information, the approach adopted for message reading was rule-based pattern-matching which serves to extract needed pieces of information from each of the telex messages and combine them for final output. This, then, required the new creation of a knowledge base incorporating telex-specific expressions, and business habits as well as the knowledge and know-how of experts. The knowledge base consists of object classes and production rules, sub-programs for message analysis, keyword files for reasoning and dictionaries of proper nouns for bank names. The scale of message-understanding ratio was also adopted for evaluation of the system's performance. The system's distinetion is in that it is a kind of document-understanding system using a knowledge bese, and in that it is a real-time expert system operating 24 hours a day.

1. はじめに

金融の国際化の進展にともない，外為取引やディーリングなど銀行における国際業

務は急伸張を見せており，海外の金融機関との資金決済電文や各種通信文のやりとりはますます増加している．通常これら海外金融機関との電文の送受信は，SWIFT* ネットワークまたは国際テレックスを通じて行われている．

このうちSWIFT電文は，コンピュータ処理可能な形式に標準化されており，その受信電文は，日銀ネットの外為円決済システムや勘定系システムに自動的に連動することができる[1]．

しかし，テレックス電文は，その特徴として，普通の手紙文と同様に非定型であること，省略の多い商業英語で書かれていること，文章表現のほか表形式や箇条書きで表現される場合があることなど，様式等が統一されていない．また誤字，脱字等タイプミスや電文の途中切断もかなりあり，正確な解読には豊富な知識と熟練が必要となっている．このため，この作業だけはシステム化が困難とされ，ベテラン行員の目による判読・仕分けを必要とし，人手の介在を余儀なくされていた．

このたび㈱三和銀行において，この国際テレックス電文の解読処理を人工知能の手法を使って自動化した．本稿では，限定的ではあるがテレックス文という自然語文をコンピュータにより解読することを実現した三和銀行テレックス電文解読システムの技術成果について，その開発を担当した立場から報告する．

2. テレックス電文解読システム

2.1 システムの目的

従来三和銀行における国際テレックス受信事務は，テレックス受信端末よりプリントアウトされた受信モニタをもとに手作業（一部パソコン利用）で行われていた．今回，同行において海外事務全般の合理化を目的に，テレックス電文の解読作業を含む一連の業務のシステム化が企画され，国際テレックス自動受配信システムが開発された．

本テレックス電文解読システムは，国際テレックス自動受配信システムのサブシステムに位置付けられる（図1）．テレックス受信電文を自動解読することにより，テストキー** の計算，照合，および受信電文の担当部署への配信という業務全体を自動化することを目的とした．本システムの主要機能は次の二つである．

1) テストキー計算要素を電文から抽出する．
2) 電文の表す取引種類を判別し配信先部署を特定する．

2.2 システムの機能範囲

1) テストキー計算要素

以下の情報をテストキー計算要素として電文中より抽出する．

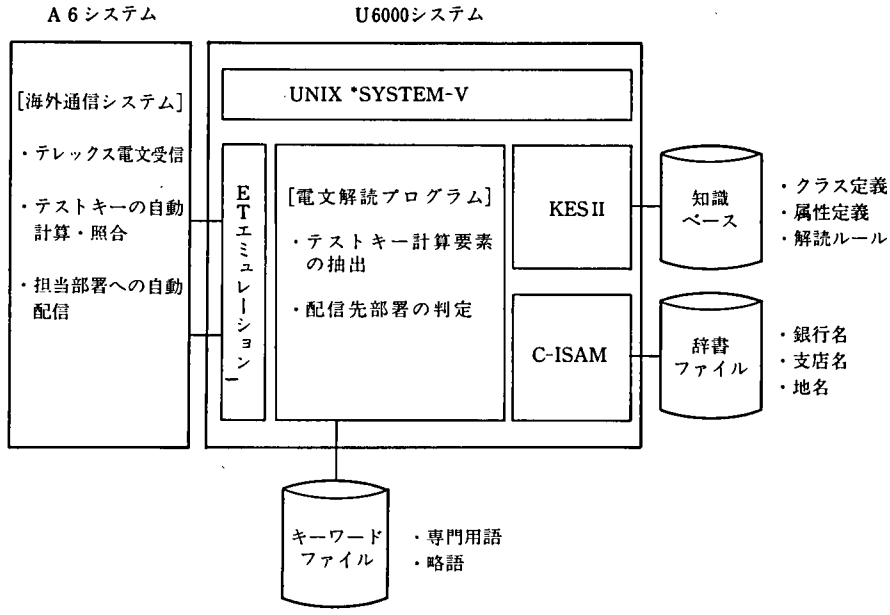
① 発信元銀行：当該テレックスの発信元の銀行名と支店名
② テストキー：電文の正当性を認証するための情報
③ テスト対象銀行：テストキー照合の相手のコルレス先*** 銀行名と支店名．

---

* SWIFT: Society for Worldwide Interbank Financial Telecommunication の略．1973年に設立された非営利法人．加盟金融機関の国際金融取引に関するメッセージ交換を行うネットワークの開発・運用を行う．

** テストキー：セキュリティを目的としテレックス電文中に挿入される一種の暗証番号．電文中の特定の日付，金額等を計算要素として銀行間で取り決めた方法で算出された値と照合される．

*** コルレス先：為替取引契約を結んだ銀行．



*UNIXオペレーティング・システムは，UNIX System Laboratories, Inc. が開発し，ライセンスしている。

図 1　国際テレックス自動受配信システムの構成

特別な指定がなければ発信元銀行が暗黙値となる．

④　テスト東西区分：テストキー照合の対象となる三和銀行の東西（東京/大阪）の別

⑤　テスト対象日付：テストキー計算に用いる日付

⑥　テスト対象金額：テストキー計算に用いる幣種と金額

2)　取引種類と配信先部署

電文の表す取引種類を導き出し，配信先部署名とその東西の別を判定する．そのために電文中より以下の項目を抽出する．

①　口座番号

②　受益者名（銀行，会社，個人名）

③　口座保有銀行名と支店名

④　電文または取引のレファレンス番号

⑤　アテンションの部署名

本システムで識別する取引種類は顧客送金，銀行間付替，信用状の開設等現在のところ 22 種類，そして配信先部署は東西合わせて 18 部係である．

3)　そ の 他

①　1 電文中に複数の取引が含まれている場合がある．その認識と取引の切り分けを 5 取引まで行う．

②　"AVOID DUPLICATION"の表現（二重処理への注意メッセージ）を電文中から抽出する．

③　不完全電文（途中で切れている電文）を一定の条件下で識別する．

④　「訂正」への対処はとくに行わない．

2.3 電文解読ノウハウのシステム化

テレックス電文の解読とは，電文中から必要な項目を抽出することと，それをもとに電文を分類することである．そのためのアプローチにはいくつかの方法が考えられる．電文の内容を英文法と辞書をもとに1単語ずつ解釈していく方法は，機械翻訳と同様に自然言語処理の領域である．この方法の長所は，きわめて汎用的で誤りも少ないと考えられることである．反面，技術的に非常に困難であり，実現できたとしても解読処理に長時間かかるものと予想される．もちろん，本システムの目的は電文全体を翻訳することではない．

本システムでは，ある種のエキスパートシステムとして本アプリケーションをとらえた．すなわち，要求される出力情報が特定されていることに着目し，パターンマッチング手法を用いて出力に必要な情報を個別に電文中から抽出するとともに，それらを組み合わせ，業務固有の規則をもとに最終的な出力結果を得るという方法である．解読の精度には限界があるが，長所として，処理効率がよい，英語以外の言語にも対応できる，柔軟性に富み拡張性・保守性にすぐれると考えた．米国で概念依存表現や事例ベース推論を利用した同じアプリケーションの開発事例が発表されている[2][3]が，われわれは電文解読のためのノウハウを蓄えたルールベースとそれをもとにした後ろ向き推論を基本にシステムの構築を試みた．なお，われわれと類似のアプローチでの開発事例が日本の他の銀行において報告されている[4]．

開発にあたり，過去の電文の調査・分析を行い，同時に業務担当者からの知識獲得を実施した．そして，テレックス電文の構造や表現法，国際業務における商習慣，さらには業務担当者のヒューリスティックな知識やノウハウを知識ベース化した．

知識ベースは以下のものから構成される．

1) オブジェクトクラスとプロダクションルール……電文解読のための規則を表現する．IF～THEN 形式で表現できる知識をプロダクションルール(以降ルールと記す）として蓄積し，推論中にルールからアクセスする電文構成要素や推論結果等の情報はオブジェクトクラス（以降クラスと記す）として定義する．狭い意味での知識ベースである．

2) 電文解析サブプログラム……電文構造の解析や，特定の形式に従がった項目の抽出などルールでは表現できない手続き的な知識をC言語で記述し，機能ごとにモジュール分けしている．

3) キーワードファイル……業務関連の専門用語や略語等のキーワードは電文解読のための手がかりとなる重要な情報である．キーワード情報はキーワードファイル上に保持し，実行時にメモリ上に二分木に展開し検索に使用する．現在133種類のキーワードをもつ（図2)．

4) 辞書ファイルとマスタファイル……電文中に現れる銀行名や支店名などの固有名詞を識別・抽出するためのISAMファイルで，名称を辞書ファイルに，そのマスタ情報をマスタファイルで保持・管理する．銀行名辞書，支店名辞書，地名辞書の3種類の辞書と，コルレス先銀行マスタ，国内銀行マスタ，受益者マスタがある（図3)．正式名称の他，略称や別名も同時に登録される．

銀行名のキーワード	信用状の開設を表す動詞
BANK	OPEN
BK	OPENED
BANKING	ISSUE
BANQUE	ESTABLISH
BANCO	FORM
BANCA	ADVISE
BANKEN	NOTIFY
CREDITO	・
GINKO	・
SHINYO KINKO	・
・	
・	

図 2　キーワードの例

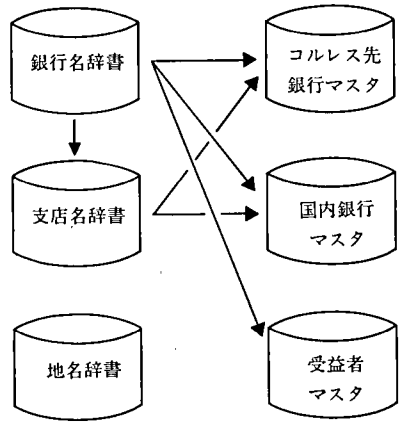


図 3　辞書ファイルとマスタファイル

3. 電文の解読

3.1 解読処理の流れ

テレックス電文の解読処理の全体的な制御は，電文解読プログラムが行う．電文解読プログラムは，電文を1件受信し，それを解読して結果を送り返す．この一連の処理を繰り返す．また，解読結果の記録，および解読処理のトレースの機能をもつ．

図4に1電文ごとの解読処理の流れを示す．図の左側は電文解読のために参照される知識の種類を，右側は処理の中間結果を表す．

3.2 電文構造の解析

テレックスの電文本体をその構成要素に分解する．合わせて，分解した各要素の属性を設定する．構成要素としては，意味を有する最小の単位である「語」のほか，「行」，「ブロック」，およびヘッダ部，本文部，トレーラ部からなる「部」がある（図5）．これらはすべてクラスとして定義されており，属性として内容そのものの他，電文中での位置を表す情報等をもつ．

1)　語

1個以上連続した「区切り文字」（空白，コンマ，ピリオドや改行文字等）で分割されたものを「語」と呼ぶ．英字部分と数字部分からなる語の場合は，英字部

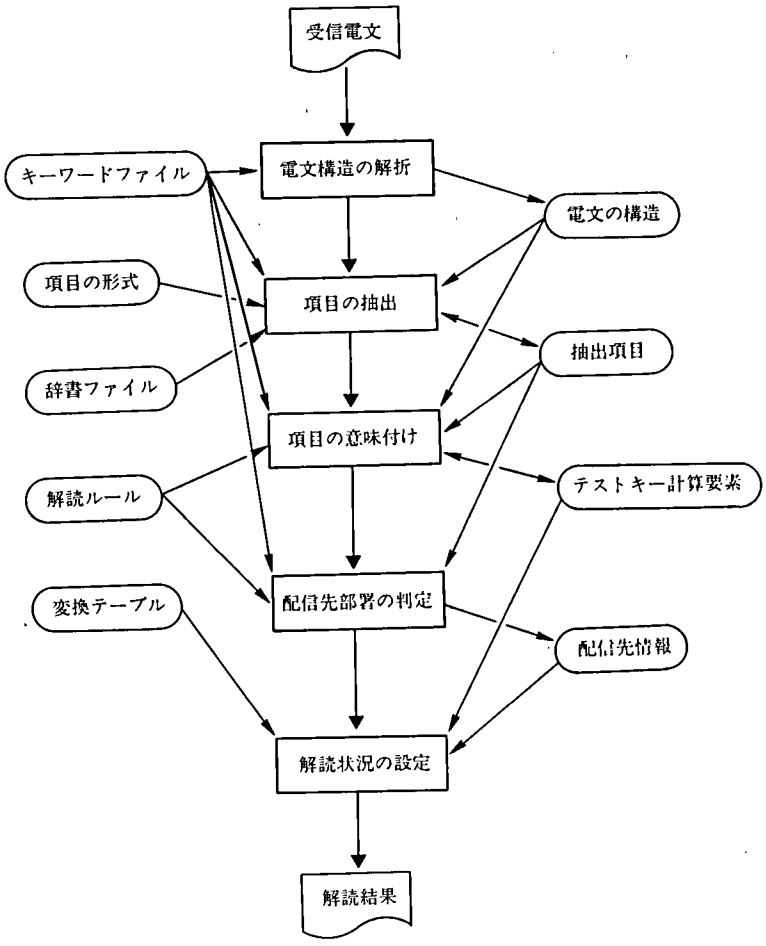


図 4　解読処理の流れ

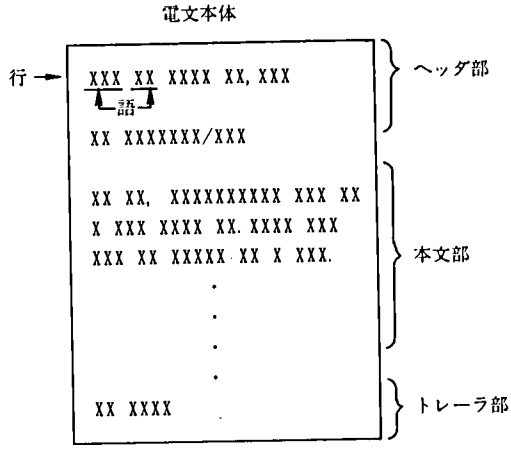


図 5　電文の構造

分と数字部分を独立した語として分離する．

逆に，区切り文字で分けられた連続する語が，次に示すような意味的に一つにまとめるべきものであると判断するならば，それらの語をすべて結合して（場合によっては区切り文字も含めて）一つの語として扱う．

① 熟語:複数の語からなるキーワード
　ex. 「CREDIT ADVICE」「IN FAVOR OF」「A/C NO」
② 略語:「英字一字」「ピリオド」「英字一字」「ピリオド」の形式をしたものは，ピリオドを取り除き，2文字を結合し1語とする．
　ex. 「N.Y.」→「NY」「H.O.」→「HO」
③ 略語:略語用の区切り文字（スラッシュ，ピリオド，アポストロフィ）を含みキーワードとして登録されたもの
　ex. 「INT'L」「L/C」「I.F.O」

語は以下のような属性をもつ．

① 語の内容
② 語通番（電文本体の全体での通し番号）
③ 語の位置する行番号
④ 語の行内の順序番号
⑤ 文字数
⑥ 文字種（数字，序数，その他）
⑦ 前区切り文字
⑧ 後区切り文字
⑨ 所属する部
⑩ 所属する取引番号

2) 行

語は「行」を構成する．行には，実際に語の存在する「文章行」と，語のない「空白行」とがある．本システムで単に行といえば文章行を指す．

行の区切りは，改行文字，または改行文字と復帰文字の組み合わせで識別する．

行は以下の属性をもつ．

① 行番号
② 行の先頭の語の語通番
③ 行の最後の語の語通番
④ 後続する空白行数
⑤ 所属する部

3) ブロック

空白行によって区切られた文章行の集りを「ブロック」と呼ぶ．電文中に複数の取引が存在する場合に取引の境界を識別する手がかりとなる．

ブロックは以下の属性をもつ．

① ブロック番号
② ブロックの先頭の行の行番号
③ ブロックの最後の行の行番号

④ ブロックの先頭の語の語通番
⑤ ブロックの最後の語の語通番

4) 部

「部」は電文中から必要な項目を探し出す時にその範囲を限定する役割をもつ．ただし，部の境界を明確に分けることのできない場合や，ある部が存在しない場合もある．

各部の行範囲は「本文部の始まりのキーワード」と「本文部の終わりのキーワード」という2種類のキーワードをもとに決定する．これらのキーワードが存在しない時は，あらかじめ定めた電文本体の先頭および最後からの行数をもとに仮定する．

部は以下の属性をもつ．

① 部のID（ヘッダ部，本文部，トレーラ部）
② 部の先頭の行の行番号
③ 部の最後の行の行番号
④ 部の先頭の語の語通番
⑤ 部の最後の語の語通番
⑥ 行数

5) 取引

複数取引電文の認識とその切り分けを5個まで行う．分割された各々を「取引」と呼ぶ．特定のキーワードや項目が複数個存在すること，およびそれらの属性と位置関係から取引の識別を行う．

取引は以下の属性をもつ．

① 取引番号
② 取引の先頭の行の行番号
③ 取引の最後の行の行番号

3.3 項目の抽出

抽出する項目の形式がある程度定まっているもの，あるいは辞書ファイルやキーワードファイルを検索することにより項目を取り出すことができるものは，電文解析サブプログラムの手続きによって項目を抽出する．また，項目によってはその出現する電文中の位置（ex. 行の先頭）や範囲（ex. ヘッダ部）が限定できるもの，あるいは，特定のキーワードや他の項目との位置関係を利用してその項目を抽出することのできるものがある．これらの処理により抽出された項目は，最終的に必要となる項目の候補となる．

1) 形式に決まりのある項目
- 日付
- 金額
- テストキー
- レファレンス番号

2) 辞書ファイルで検索できる項目
- 銀行名，支店名（コルレス先銀行，国内銀行）

・受益者名（会社，個人）
・地名

3) キーワードとして登録される項目
・幣種の名称
・アテンションの部署名
・"AVOID DUPLICATION"の表現

4) キーワードが手がかりとなる項目
・テストキー
・口座番号
・アテンションの部署名

3.4 項目の意味付け

抽出された項目に意味を付け，最終的に必要な項目を決定していく．項目の特定のために，電文の構造やキーワード，他の意味付けされた項目との位置関係等を利用する．このプロセスはルールによる推論処理が中心である．

項目間の関係を図6と図7に示す．

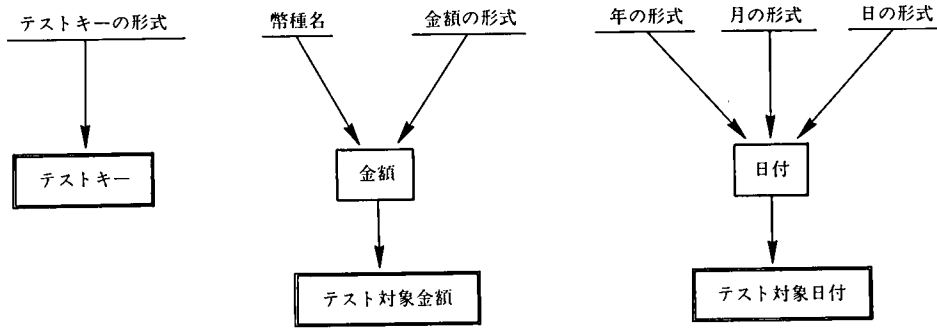


図 6 項目の抽出と意味付け−1

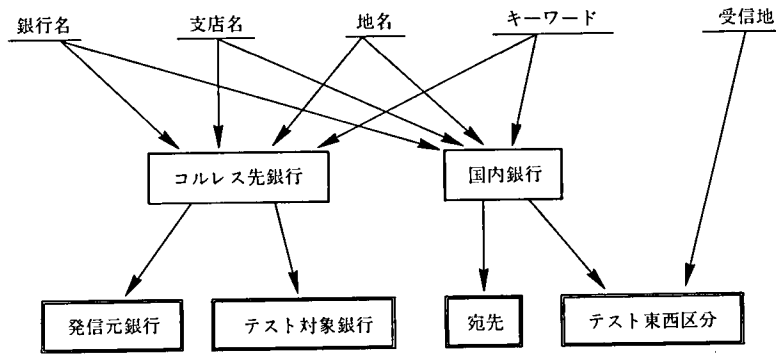


図 7 項目の抽出と意味付け−2

3.5 配信先部署の判定

電文の内容を表す取引種類を判定し，配信先部署を決定する．すでに抽出され意味付けされた項目のほか，特定のキーワードの存在，およびそれらの属性と電文中の位

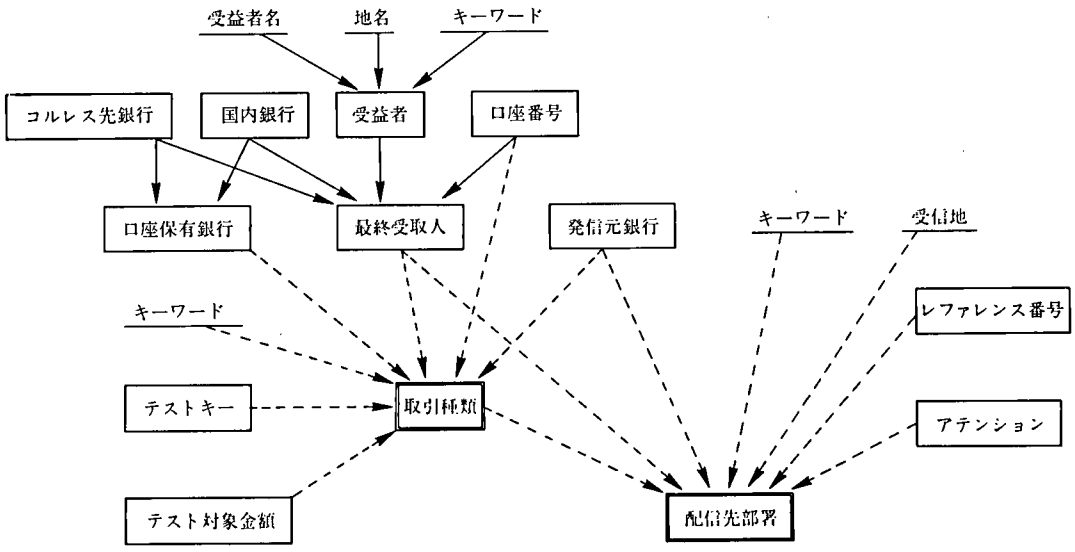


図 8　配信先部署の判定

置関係を利用する．このプロセスもルールによる推論処理が中心である．

配信先部署の判定に関わる項目間の関係を図 8 に示す．

3.6 解　読　例

非常に単純化した架空の電文ではあるが，図 9 に解読例を示す．また，この電文の配信先部署の判定に使用されるルールを以下に示す．

① ＜支払の動詞＞の後ろ 5 行以内に＜支払先のキーワード＞，＜国内銀行＞があり，さらに 5 行以内に＜受取人のキーワード＞，＜受益者＞があれば，＜国内銀行＞が［口座保有銀行］であり，＜受益者＞が［最終受取人］である．

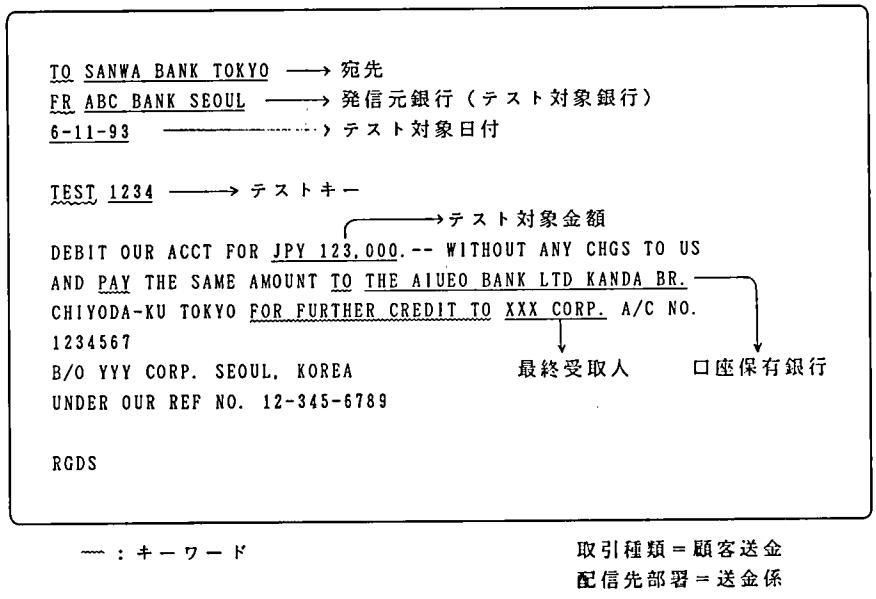


図 9　解 読 例

② ＜支払の動詞＞があり，＜最終受取人＞が銀行でもノンバンクでもなければ［取引種類］は＜顧客送金＞である．

4. 稼 働 環 境

4.1 システムの全体構成

国際テレックス自動受配信システムの全体構成を図10に示す．国際テレックス網を通して送られてきた電文は，まずテレックス受信機PC-860で受信され，電子計算センターのユニシスA6内海外通信システムに送られる．海外通信システムは2台あるユニシスU6000/55のいずれかに1電文ずつ送信し，解読結果を受け取る．そして，テ

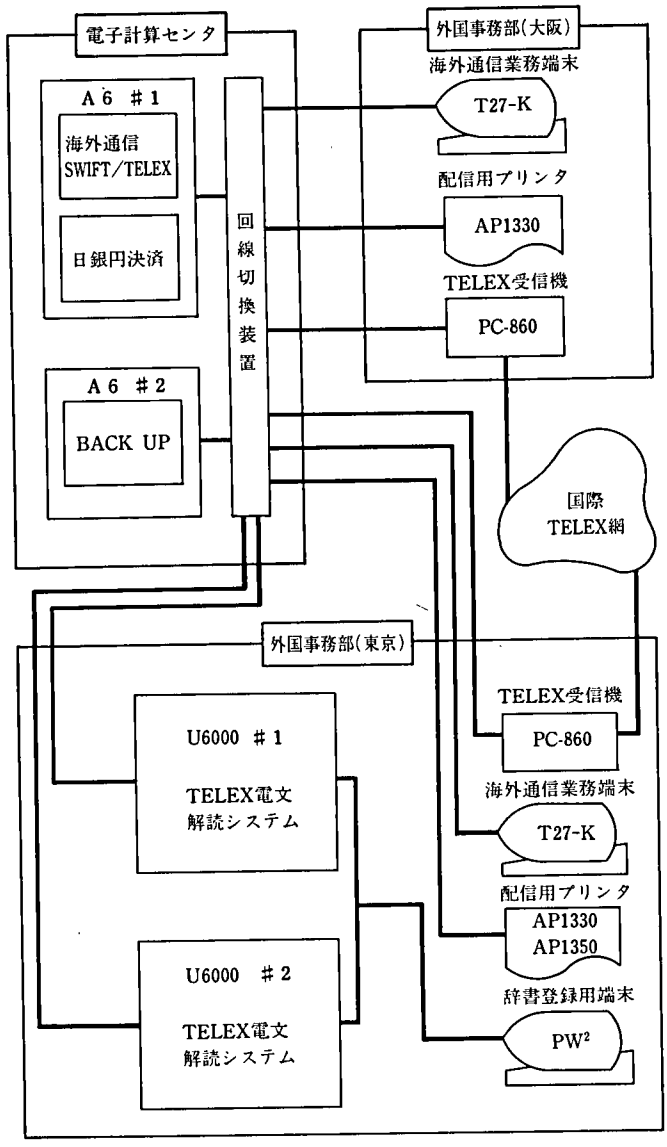


図 10 国際テレックス自動受配信システムの全体構成

ストキーを計算，照合するとともに，判定された配信先部署のプリンタに原電文と解読結果を出力する．

テレックス電文は時差の関係で昼夜の別なく海外から送られてくるので，本システムも24時間稼働とした．そのため，負荷と危険の分散を目的にU 6000/55は2台構成にし，相互バックアップと本番中の保守作業を可能にした．

U 6000システムはA 6システムのバックエンドに位置付けられ，トランザクション処理の形態は問い合わせ応答型である．受信した電文を即時に処理するリアルタイム型のエキスパートシステムであることが特徴である．

4.2 ソフトウェア構成

本システムの開発に利用したソフトウェアは次のとおりである．

1) KES II (Knowledge Engineering System II)*……プロダクションルールによるエキスパートシステム構築用ソフトウェアであり，電文解読のためのクラスとルールを蓄え，推論を実行する．
2) C-ISAM……C言語によるISAMファイルシステムの構築・運用のためのソフトウェアである．電文中の銀行名，支店名等の検索に用いる．
3) ETエミュレーション……Aシリーズホストと POLL/SELプロトコルで接続し，U 6000をホストのMT 983-JあるいはT-27 Kとしてエミュレートする．A 6システムとの間で電文の送受信を行う．

5. システムの評価

5.1 システムの規模

1994年2月現在のシステムの規模を以下に記す．

① 解読ルール数：318（ゴール数：16）　　クラス数：121
② キーワードの種類：133　　キーワード総数：1519語
③ 辞書ファイル
　　銀行名：約2800件
　　支店名：約6000件
　　地名　：約800件
④ Cプログラム
　　電文解読本体　　：約20000ステップ
　　電文解読の制御・管理：約8000ステップ

5.2 システムの性能

5.2.1 解 読 率

本システムの性能を定量的に評価するために解読率という尺度を導入した．

$$\text{解読率}=\frac{\text{完全に解読された電文数}}{\text{全電文数}}$$

完全に解読された電文とは，テストキー計算要素と配信先部署の両方とも正しく解読されたものである．

システムの開発を始めるにあたり，本番開始時の解読率として70%の目標を掲げ

* KES IIは米国Software Architecture & Engineering社の登録商標である．

た．そして生の電文を使ってテストと検証を行い，解読率を徐々に上げていった．解読が不完全であった電文には現象として「解読不明」と「誤解読」の2種類があり，その原因は以下のように分類できた．

1) 原電文自体が誤りや矛盾を含んでいる場合（人間の専門家でも解読が難しい）
2) 長電文で，システムで予め定めた解読許容時間を超えたもの（処理を中断し解読不能にする）
3) システムの対象とする範囲外の電文が入力された場合，すなわち知識ベース内の知識の不足による場合（予め想定している場合とそうでない場合とがある）
4) 本システムに誤りがある場合

このうち1)と2)はもともとの電文に起因するものであるが，3)と4)に関しては，システムの改善により比較的容易に解読率を上げることができると期待された．しかし3)については，分析の結果，解読ルールの条件がゆるいと解読を誤ることが多く，逆に条件がきびしすぎると誤解読は少ないが解読不明となるケースが多いことが判明した(図11)．また，当然のことではあるが，より厳密な解読を行うためにルールの条件の数を増やすと解読時間が長くなることもわかった．あちらを立てればこちらが立たないという状況下で，ルールの条件のきびしさをどのあたりに落ち着かせるかはまさに試行錯誤の連続であった．実際には，数千の生の電文を使って十分なテストを行うことができ，本番稼働時で85%の解読率を達成することができた．

エキスパートシステムは適用業務の性格上知識ベースの保守が重要視されている．本システムにおいても本番稼働後も，キーワードや辞書の追加登録，解読ルールの拡充により継続的に解読率の向上を図っている．ただ，解読率の向上とは別に，電文(取引)の重要性に鑑みて，誤解読が極力少なくなるように考慮するとともに，最終的には配信先部署の行員が内容をチェックするという形をとっている．

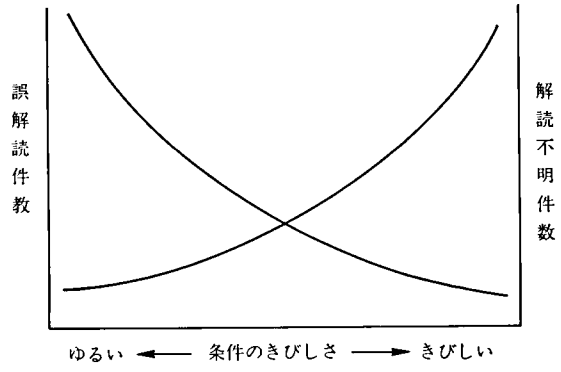


図 11 解読精度の分析

5.2.2 応 答 時 間

解読時間，すなわちA6システムへの応答時間については，U6000/55二台で10時間に1000件の電文を処理しなければならないと想定し，1電文あたり1.2分を当初の目標とした．また，A6システムに対する応答を保証するため，前述のとおり解読許容時間（現在は8分）を設定し，これをこえた場合はその時点で処理を中断し解読不能

として返すことにした．

しかし，開発途中のプロトタイプでは，少々長い電文になると数十分もかかるという状況であった．分析してみると，推論の実行に最も時間を要することが判明した．すなわち，ルールの条件部で参照するクラスのメンバの組み合わせの数が多すぎるためであることがわかった．そこで，電文解析サブプログラムで推論のためのメンバ作成の条件をきびしくすることにした．さらに，ルールと推論方法の見直しと改良を行い，本番稼働時で1電文あたり平均30秒の応答時間を得ることができた．

6. おわりに

本稿では，知識ベースを利用したテレックス電文の自動解読システムについて，その機能と解読メカニズムについて述べた．業務領域固有の知識を利用することにより，限定的ではあるが効率的かつ保守性にすぐれる実用的な文書理解システムを構築することができた．

三和銀行では，すでにユニシスA6をホストとする国際通信システムIBS/TIPSS[5]を導入して国際通信事務の機械化を図っていたが，これに今回のテレックス電文解読システムを中心としたテレックス自動受配信システムの運用を付加することによって国際通信事務全体の自動化が達成された．

本システムは1991年11月に本番を迎え，その後自動配信先部署の追加などシステム範囲の拡大が行われた．本システムの稼働により，顧客サービスの向上，事務の省力化などの効果が得られている．

末筆ながら，本システムの開発ならびに本稿の執筆にあたりご指導，ご協力いただいた方々に感謝の意を表する．

参考文献

［1］ 渡辺実，”国際金融通信ネットワーク「SWIFT」の機能とサービス”，Systems，ユニシス研究会，No.243，1992年3月，pp.41～46.

［2］ S.L.Lytinen and A.Gershman，”ATRANS: Automatic Processing of Money Transfer Message”，Proceedings of AAAI，1986，pp.1089～1093.

［3］ M.Goodman，”Prism：A Case-Based Telex Classifier”，Innovative Applications of Artificial Intelligence 2，AAAI/MIT Press，1991.

［4］ 橋本諱嗣，”住友銀行のAI適用事例”，金融情報システム，(財)金融情報システムセンター，NO.57，1988年12月，pp.49～55.

［5］ 定免貞比古，渡辺実，栗原毅，”金融機関の国際業務を支えるソリューション・システムIBS”，ユニシス技報，No.34，1992年8月，pp.63～83.

執筆者紹介　中　田　純　一（Jun-ichi Nakada）

1952年生．1974年京都大学理学部卒業．同年日本ユニシス(株)入社．客先教育および都市銀行の情報系/対外接続系システムの開発に従事．現在 システム技術第一本部 知識システム部に所属，主にAI応用システムの開発を担当．人工知能学会会員．

日刊スポーツ新聞社における「案内広告システム」

A "Classified Ad System" at Nikkan Sports Newspaper

近藤千秋

要約 新聞広告は新聞社にとって，購読料と並ぶ2大収入源であるが，テレビ・雑誌・ミニコミなどの他メディアとの競合力は長期低下傾向にある．今般の広告不況でも他のメディアに先駆けて新聞広告がいち早く落ち込み，落ち込み幅も大きいことから強い危機感を持っている．日本新聞協会では1993年3月に対策プロジェクトを発足させ，技術革新の遅れている広告分野の電子化に取り組み始めた．

この分野では，先駆的に取り組んできた日刊スポーツ新聞社では，1993年2月にプロジェクトの方針を先取りする新しいコンセプトに基づく「案内広告システム」の稼働を開始した．本システムは同社と広告代理店とをオンライン接続して，EDI(電子データ交換)による案内広告の集配信を実現し，さらに迅速性が求められ新聞制作上のネックとなっていた配列・割付工程をAI(人工知能)を活用して完全自動化を実現している．これにより，案内広告掲載の業務全般の効率化・省力化が図られ，業界内の注目を集めている．本稿では，その実現に至る経緯と技術的先進性について紹介する．

Abstract To newspaper publishers, newspaper advertising is one of the two big income sources comparable to earnings from subscribers, but its power to compete with other media such as TV, magazines and 'mini' media has been on the downward path year after year. The 'advertisement' recession now under way indicates a sharp drop in the volume of newspaper advertising, ahead of the other media ads, to such an extent that it causes people in the industry to have a strong feeling of 'emerging crisis'. The Japan Newspaper Society, which launched its 'what-to-do' project in March 1993, started to work on how to electronize advertisement business, where a delay in technological innovations stands out.

Nikkan Sports Newspaper, who had taken the lead in dealing with this computer application area, started operating its own "classified ad system" based on the new concepts (forerunning the project's guidelines) in February, 1993. Linking the headquarters office on-line with ad agents, the system has paved the way for implemented ad collection and delivery with the help of EDI (electronic data interchange), and also, to meet requirements for higher rapidity, for full-scale automation of arrangement and layout which used to be a bottleneck of newspaper production with the use of AI (artificial intelligence) technology. The new system, as a proven useful tool to support more efficient ad-printing operations with less manpower, is now what is drawing the industry's whole attention. This paper describes how the system has been created and how advanced it is in technology.

1. はじめに

新聞制作におけるコンピュータの利用は編集制作分野ではCTS (Computerized Typesetting System) と呼ばれる電算写植システムがめざましい発展を遂げてきた．しかし，新聞広告分野は技術革新の波に乗り遅れ，システム的には10年以上の遅れをきたしていると指摘され，いろいろな局面において発想の転換を迫られている．今般の広告不況および新聞広告シェアの長期低下傾向(過去20年で10%以上)に対する危

機感から日本新聞協会・広告委員会では1993年3月に「新聞広告の電子入力に関する研究会」を発足させた．当研究会では文字広告の集稿・管理業務のオンライン化を目指し，広告代理店をも含む業界全体の問題として改革に取り組んでいる．改革のステップは，文字広告の代理店入力，ディスプレイ広告の直接入力，テレ・マーケティングの導入の3段階を想定している[1]．

日刊スポーツ新聞社では従来から広告分野におけるコンピュータ利用では先駆的に取り組み，1990年に「広告割付システム」，1993年に「案内広告システム」の稼働を開始している．「案内広告システム」は代理店側の案内広告原稿作成から媒体(新聞社)側の紙面掲載に至る全ての作業を電子化・オンライン化したものである．案内（文字）広告の代理店入力，掲載料金の計数管理を実用化し，研究会の提言する改革ステップの第一段階に到達している．本システムは電子入力だけでなく，媒体側においても処理（原稿管理・物件配列）の自動化を図り，「省力」「迅速」「正確」に抜群の効果を発揮している．

2. 案内広告システム

2.1 システムの概要

本システムは，新聞制作工程において案内広告を対象として原稿の作成から割付(配置）までを分担する．図1に統合新聞制作システムの全体像を示す．一般的に新聞社では自社原稿を扱う部署（編集局）と広告原稿を扱う部署（広告局）は別組織であり，紙面の制作責任をあらかじめ分担し，独自に編集責任を負う．

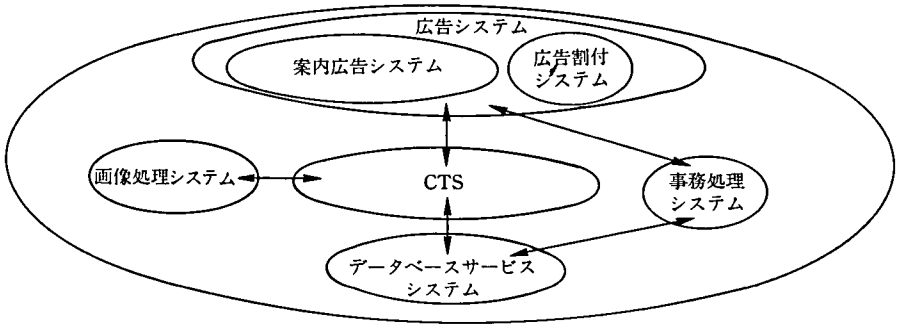


図1　統合新聞制作システム

日刊スポーツ新聞社における広告システムは「広告割付システム」と「案内広告システム」から構成される．「広告割付システム」は広告原稿の紙面掲載位置を決定するシステムであり，編集局と広告局の責任範囲を分離する．該システム上では2行物～138行物にわたる案内広告物件は数百件の掲載対象の集合体を一つの広告として認識する．これは案内領域の大きさと掲載位置が空白の状態で確保されたことを意味する．図2は広告割付の操作画面の例である．「案内広告システム」は広告代理店で制作した案内広告原稿を集稿して，案内領域に見栄えよく自動的に配列するシステムである．

本システムは三つのサブシステムから構成される．図3にシステム構成を示す．

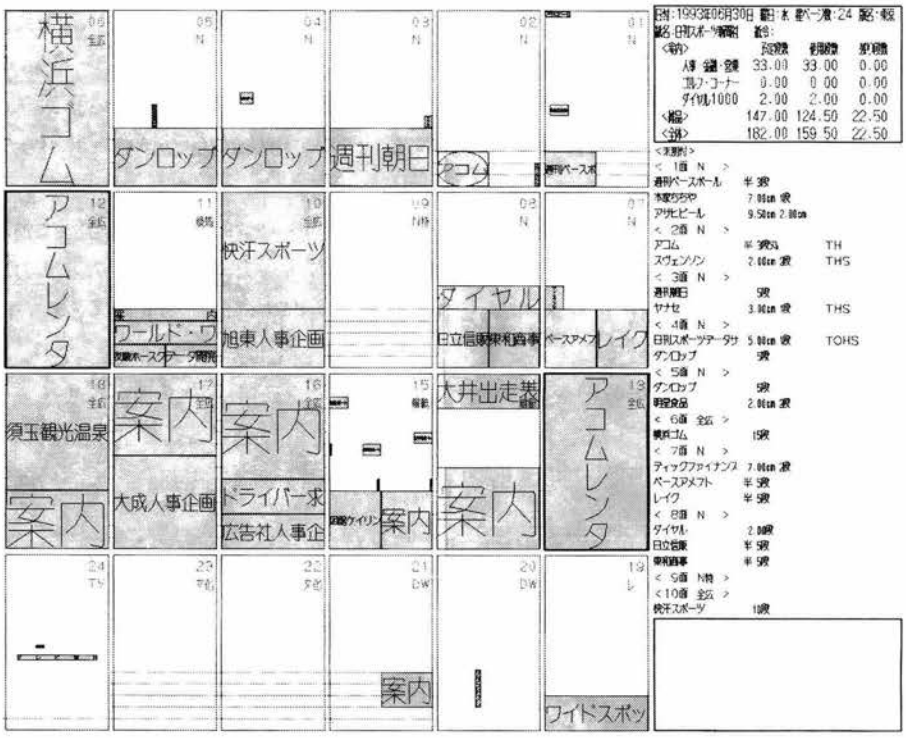


図2　広告割付の操作画面

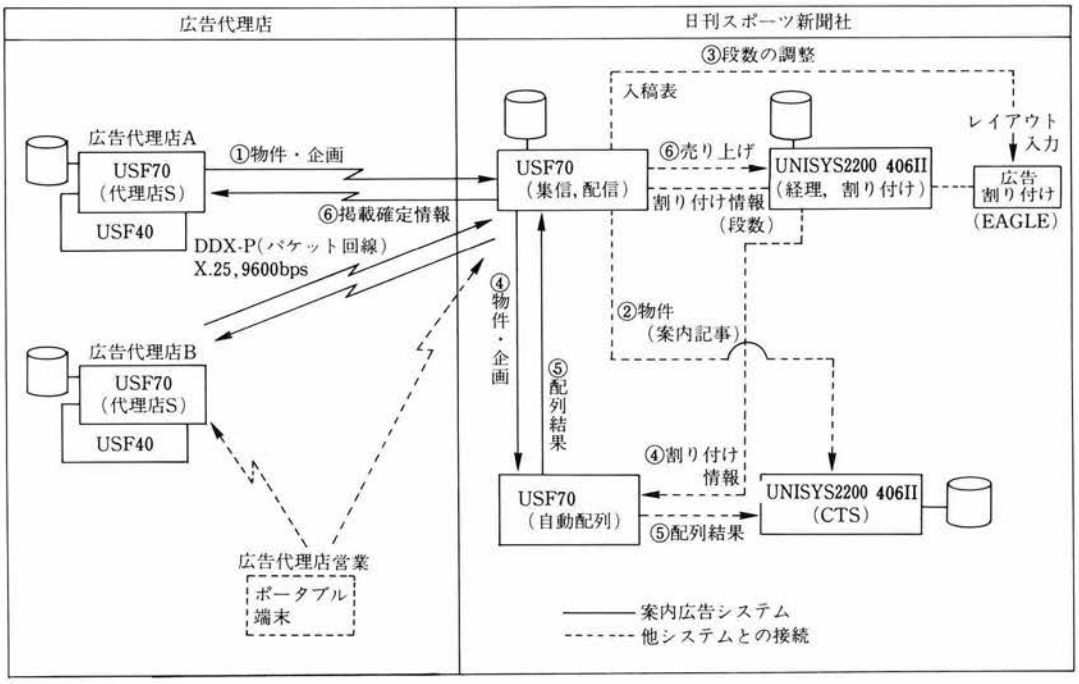


図3　案内広告システムのシステム構成図

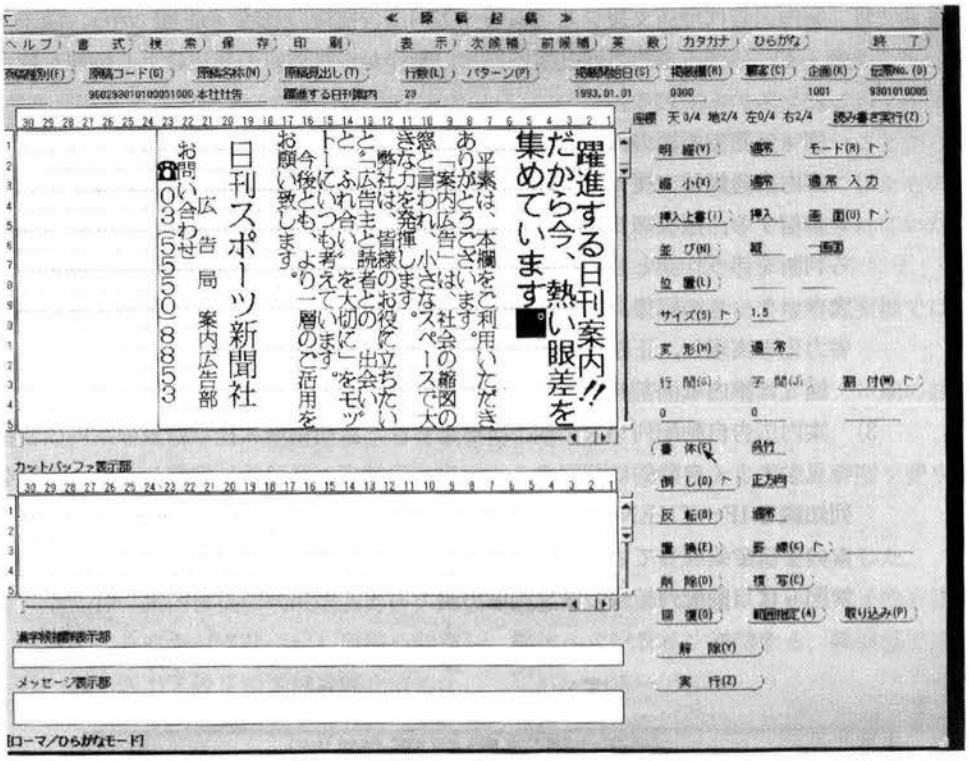


図4　案内原稿の入力画面

入稿表

対象　：　93年 05月 31日　　出力　：　93 年 06 月 01 日　10 時 43 分

項番	掲載欄コード	掲載欄名	件数	行数	段数	換算段数	EAGLE段数	差引段数
1	1000	募ね人	0	0	0.000			
2	1110	理容	0	0	0.000			
3	1120	美容	0	0	0.000			
4	1230	予備	0	0	0.000			
5	2000	募集（女子）	2	7	0.051			
6	2110	飲食・食品	11	35	0.254			
7	2120	一般	39	126	0.913			
8	2130	建築・土木	46	131	0.949			
9	2140	電気・技能・生産工	13	63	0.457			
10	2150	計員・営業・事務	18	304	2.203			
11	2160	新品販売店	84	524	3.797			
12	2170	運転手・自動車・整備	11	101	0.732			
13	2180	ハイヤー・タクシー	28	367	2.659			
14	2190	パチンコ	10	142	1.029			
15	2310	警備	0	0	0.000			
16	2320	その他	0	0	0.000			
17	2400	募集（計交）	0	0	0.000			
18	2500	募集（男子）	8	27	0.196			
19	2600	パート・アルバイト	1	10	0.072			
20	2700	中高年	0	0	0.000			
		人事合計	271	1837	13.312			
21	3000	不動産	0	0	0.000			
22	3110	金融	80	504	3.652			
23	3120	マネーショップ	44	1770	12.826			
		金融合計	124	2274	16.478			
24	3200	自動車	0	0	0.000			
25	3300	営業	3	6	0.043			
26	3400	学校・教授	0	0	0.000			
27	3500	病・医院	0	0	0.000			
28	3600	雑件	0	0	0.000			

終了

[無変換]

図5　入　稿　表

1) 案内広告代理店支援システム……案内広告物件の受注（売上）から，請求・入金・支払に至る事務処理と案内原稿の制作から媒体への発注に至る処理を統合化したシステムである．

図4は案内原稿の入力画面の例である．

2) 案内広告媒体支援システム……各代理店からの受注情報・原稿情報をオンライン集信する．原稿のライフサイクル管理・集計業務は自動化された．人間系による判断を伴う作業として原稿審査・案内紙面割り調整（原稿量と案内領域の差が大きいケースの編集局・広告局間の調整作業）を残すが，媒体管理業務は格段の省力化を実現し，正確性・迅速性を向上させている．

図5は案内紙面割り調整用の入稿表の出力画面の例である．

3) 案内広告自動配列システム……与えられた案内領域へ掲載対象の案内広告物件を見栄えよく自動的に配列する．従来，熟練者が経験的に把握していた煩雑な配列知識をIF～THEN～のルール形式で知識ベースに蓄え，エキスパート・システムの手法で処理している．

図6は自動配列結果の検証画面の例である．

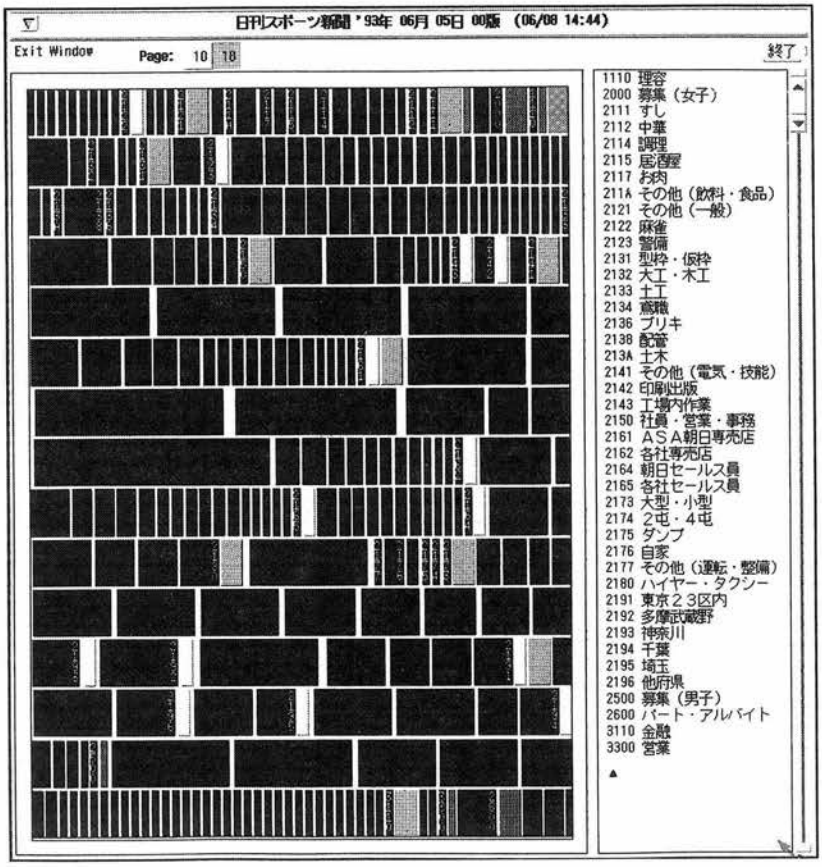


図6　自動配列の結果表示

2.2　システム導入の背景と目標

図7はシステム導入前後の業務フローの記述である．導入前は手書きの申込書をFAX受信した後，新聞社の関連企業のパンチ入力部門で原稿を作成し，赤字が無くなるまで校正を繰り返し，完全原稿としていた．求人募集の構成比が高いことから，電話番号・給与額の校正ミスは許されない．配列システムは機械化されていたものの，結果入力のシステムであり，配列作業そのものは人間の頭の中で実行されていた．当時の状況認識として

1)　案内原稿のライフサイクル管理が複雑で，FAX・電話による情報交換では転記ミス・入力ミスが多い．

2)　媒体側における外部委託費（原稿入力・配列作業）の増大，バブル期の好景気を反映した入稿量増による管理業務の増大

3)　熟練担当者による経験と勘に依存した配列作業は多大の労力と時間を費やす．

等が挙げられる．

このような状況を解決するシステムの目標として下記の4項目を設定した．

1)　案内原稿の代理店入力……原稿の発生源である代理店において電子的方法により完全（クリーン）原稿を制作し，媒体へオンライン送信する．媒体側では原稿入力・校正の工程は廃止する．

2)　案内原稿のライフサイクル管理の自動化……自社原稿は作成～掲載～破棄の一過性のライフサイクル管理で十分であるが，案内原稿は複数回の連続掲載，特定日（たとえば土日祭日）の連絡先の変更等の掲載内容の切り替え，追加掲載，掲載中止および継続掲載の見込み広告主の原稿保存等のライフサイクル管理を必要とする．代理店から媒体への発注時指示により媒体側でのライフサイクル管理は自動化し，掲載ミスを防止する．

3)　ワープロ感覚による原稿作成……案内広告原稿の制作は入力の専門家にとって

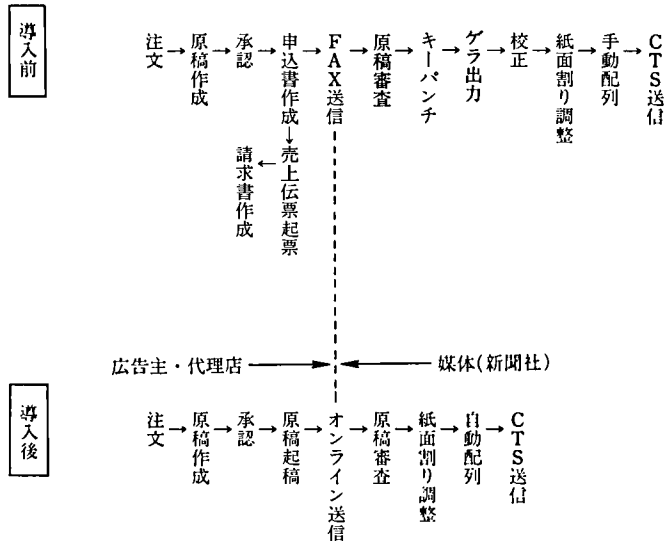


図7　システム導入前後の業務フロー

も一般記事の制作と比較して熟練した技術を要求される．しかし代理店入力は新システムの大命題であり非専門家であっても簡単な操作で原稿を作成できる．

4) 配列時間の短縮……旧システムでの配列作業の所要時間は2〜3時間/日であり，特定の熟練者でないとできない．新システムでは熟練者の経験と勘に頼るシステムから脱却すると共に，配列時間を大幅に短縮して原稿締切時間の延長等の顧客サービスの向上を図る．

2.3 システム化の技術的ポイント

前記の目標を具現化する方策を検討した結果，運用面を除いて技術的な課題としては次の項目に集約された．

1) 原稿入力操作は非常に簡単であり，新聞掲載と同一体裁で表示/印書されること．

2) 自動配列の実行時間は確実に一定時間内に完了し，配列結果の品質（見栄えよい配置）は実用に耐え得ること．

各々，専用ワープロ（DTP）の開発，AIツール（GNOSIS-II）を利用したエキスパート・システムの構築により実現を図る方針を決定した．

2.3.1 専用ワープロ（DTP）

一般的なワープロ・ソフトでは案内広告原稿の作成は困難である．図8は一番単純な2行物の原稿であるが，いくつかの点で違いが指摘できる．

- 縦書き/横書き混在の文章である．
- 分かち書きの文章である．
- 文字サイズの変倍（長体・偏平）の自由度が要求される．
- 行の概念は本質的に異なる．ワープロ・ソフトでの行幅は可変幅であり，行内の文字のサイズ変更により変動する．案内広告では下敷きとなる原稿用紙は固定であり，操作により変動はしない．

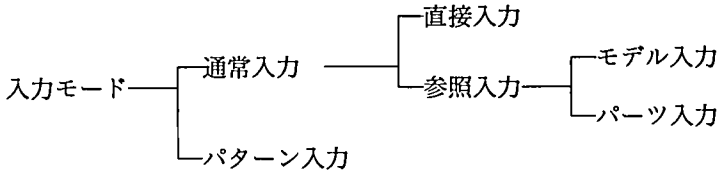
図8 2行物の案内原稿

相違点の分析から専用ワープロは文章の作成装置というより，可変サイズの文字のレイアウト装置であるという認識に立って設計している．表面上の操作はワープロ的であり，カーソルの形状は文字サイズを表現し，固定的な原稿用紙の升目の任意位置に移動可能である．文字入力は標準のカナ（ローマ字）漢字変換機構を利用している．入力された連続文字はカーソルの初期位置を起点として，カレントの文字サイズ・文字間・並び（縦/横）方向に従って，次の文字位置を連続的に決定していく．当然，用

紙大きさを超過するあるいは既に文字が配置されている等の理由により文字配置が不可の場合には，カレントの並び方向・行間から次行の開始位置を決定して上記の処理を繰り返す．

提供する機能は最新のワープロ・ソフトに比較してかなり限定されたものであるが，複雑な案内原稿の作成に必須の機能に絞り込んでいる．

入力方法として四つの入力モードをサービスする．



1)　直接入力……文字入力はキーボードからカナ（ローマ字）漢字変換で入力し，編集方法はマウス操作で指示する．画面の原稿用紙に直接，文字を埋め込む．

2)　モデル入力……保存原稿はすべてモデル入力の対象となる．体裁の似通った原稿を再利用して，一部分の変更入力により入力負荷を軽減する．

3)　パーツ入力……パーツ登録された部品原稿を利用する．

4)　パターン入力…事前に登録した体裁のパターンを利用して，文字入力のみで自動編集を実施する．モード切り替えにより通常入力モードに遷移する．

図4は直接入力モードの画面であり，左側が原稿入力部(原稿用紙に相当)，右側が編集方法の指示部である．

編集機能の詳細は割愛するが，機能の一覧のみを以下に記す．

- 並び方向　：縦書き，横書き
- 行間/字間
- 書体　：明朝，ゴチック
- 文字サイズ
- 変形文字サイズ　：長体（洋数字)，偏平（和数字）等
- 倒し　：90度単位の文字回転
- 反転　：白抜き文字
- 罫線，ブレース，パーレン　：ブレースは {}，パーレンは ()
  任意サイズの指定可
- 割付編集　：文字ブロック単位の均等配置，中央配置，上下寄せ配置等の編集
- 文字列置換
- 削除/取り込み&複写　：カット/コピー&ペイスト機能
- 文字ブロック表示
- 縮小体裁確認
- 文字合成
- カーソル移動
- 挿入/上書き
- 回復

専用ワープロの開発では新聞掲載と同一体裁で表示/印書される機能としてWYSIWYG (What You See Is What You Get) 機能を提供しなければならない．通常は同一システム内の画面と印書装置の出力体裁の一致を満たせばよいが，本システムは新聞制作工程の中間工程であり，CTS にとっては入力サブシステムに位置付けられるため，体裁の一致は CTS の最終出力である版下 (印画紙またはフィルム) までを包含する．

CTS の入力形式は日刊スポーツ新聞社独自の手続き型マークアップ言語である．これは文書 (文字列) に対して作動する操作 (文字サイズ設定等) や属性 (明朝/ゴチックの書体等) を必要箇所に埋め込む方法であり，この手続き宣言語を CTS ファンクションとよぶ．現在，約 200 種類の CTS ファンクションを保持する．図 9 は図 8 の案内原稿の CTS 入力形式である．案内広告システムの原稿情報の内部表現形式は，本質的には CTS ファンクションを解析した結果の状態を表現している．CTS ファンクションの生成処理はあたかも実行モジュールからプログラム言語を生成するような逆変換処理である．CTS との体裁の一致は重要な問題であるが，日刊スポーツ新聞社独自のローカルな問題でありこれ以上言及しない．

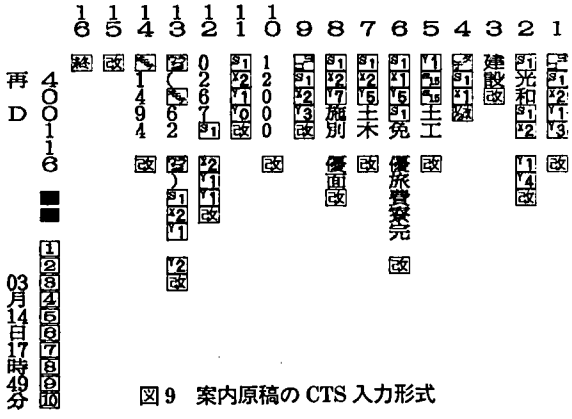


図 9　案内原稿の CTS 入力形式

本システム内での画面と印刷の出力体裁の一致を実現する手段について述べる．図 10 は印書の出力体裁を示す申込票である．

本システムで使用する表示装置 (VDT) と LBP 印書装置は弊社の標準商品である．物理的には VDT・LBP ともにラスター・スキャン機構を保持するハードウェアであり，最終的には受信 (生成) したビット列をライン・バイ・ラインに走査して出力する．それでは同一体裁にするには，如何にしてビット列を生成すればよいのか．ビット列を生成する機構を RIP (Raster Image Processor) という．RIP は文字コード・位置・サイズを入力としてビットマップを出力する．ただし，位置・サイズは出力装置の解像度を配慮して設定する必要がある．たとえばビット数が 100×100 のビットマップを解像度が 50 DPI と 100 DPI の出力装置に送信すると 2 インチ正方と 1 インチ正方の像が得られる．LBP は一般的に印書装置内に RIP を持つが，ホスト・システム内で持つ表示用の RIP とは機能仕様は完全には一致しない．さらに，文字種・文字フォントについても同様である．本システムではアプリケーション層において出力装置

申込票

日	管理番号	伝票No	掲載欄コード	行	cm	段数	回数	原稿コード	受注種別	01-01
16	1529	9304120006	2180	20	5.58	1	7	42019304120006 3001	改稿	

掲載 開始日	掲載 終了日	掲載日	04月	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
93/04/12	93/04/30		月	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31

代理店コード	代理店名	顧客コード	顧客名	件名	行数割引き	特値
4201	日刊スポーツ広告社		キャピタルグループ	0120-15-5562		

受注日	NEPSIIIコード	物件種別	版種別	企画コード	単価	手数料率	備考欄
1993/04/15	0116B047210	案内物件	日刊スポーツ全国版	一般			

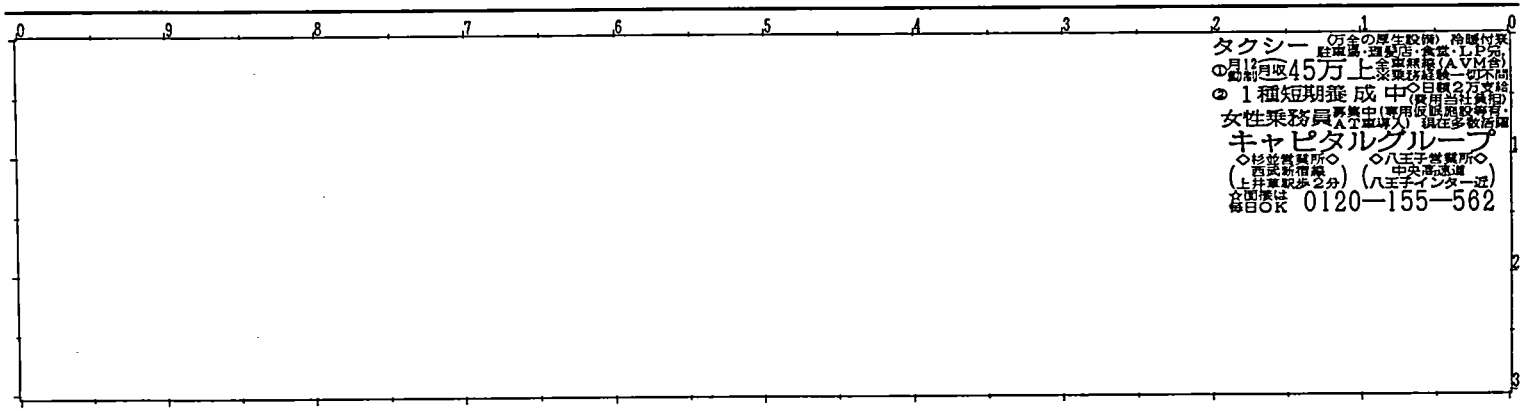


図10 申込票

の解像度を考慮したビットマップを生成する共用 RIP の開発により，画面と印刷で提供する文字の文字数・文字品質および出力体裁の完全なる一致を保証している．

共用 RIP では初期宣言において出力解像度とローカル座標系（案内広告システムで使用する単位系）を宣言し，装置からの独立を図っている．任意文字サイズの出力を前提とするため，文字フォントはアウトライン・フォントの使用は絶対条件である．ビットマップ・フォントでの文字拡大は旧式のワープロのようなギザギザ状の文字品質しか保証できない．幸いなことに CTS ではすでにアウトライン・フォントとラスタライザを使用しており，本システムではこれを流用している．このアウトライン・フォントは直線と曲線で構成され，曲線表現はベジェ曲線である．ラスタライザはスキャン・コンバージョンのアルゴリズムを採用し，ランレングス形式で出力する．

すでに RIP の中核となるフォントとラスタライザが存在するため，共用 RIP の開発は開発者の使い勝手の良さとラスタライザでは支援しない機能の提供である．共用 RIP は 20 関数を提供するが，二つの関数についてのみ説明する．説明用の仕様であり，実際とは若干相違する．

1) 文字合成（位置，サイズ，文字コード 1，文字コード 2）

共用 RIP 内で保持する文字合成テーブルでは，合成対象文字と被合成文字の相対的な挿入位置と大きさが定義されている．合成処理は次の手順に従う．

① 文字コード 1 を指定サイズで変倍計算し，ラスタ化する．

② 文字合成テーブルを検索して，対象テーブル値を入手する．

③ 文字コード 2 を指定サイズと被合成文字の相対情報から変倍計算し，ラスタ化する．

図 11 はこの手順を図式化している．

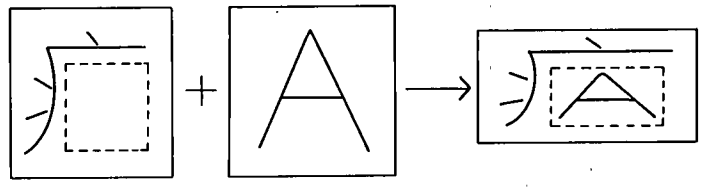


図 11　文字合成の生成手順

2) 罫線（罫コード，罫本数，罫位置×本数，罫幅）

罫線の生成は連続生成とコーナー・カットを考慮しなければならない．罫文字のフォントを単純に変倍すると，実線以外の罫は意味をなさない．たとえば，点線の拡大は線と空白を拡大して点線には見えない．罫の角（コーナー）を単純に二重表示すると罫は交差して見える．L 字型の罫線処理は次の手順に従う．

① 二つの長方形の重なり部を罫の文字サイズとする．

② 重なり部を除いた長方形の長さを文字サイズで割り，商と余りを計算する．文字サイズ×商の長さ分を中央部，残りを端部とする．各長方形は重なり部・中央部・端部に 3 分割される．

③ 中央部は罫文字のフォントを繰り返しラスタ化する．ただし，横方向のラス

タ化は90度の回転角度を与える．

④　重なり部と端部は同様の考え方で処理をする．両部はラスタ化されたビットマップをそのまま使用すると2重表示，罫の余分な延長を発生させる．そこでビットマップの一部分のみを有効とするクリップ処理を施し，スムーズな接続を実現する．

図12はこの手順を図式化している．

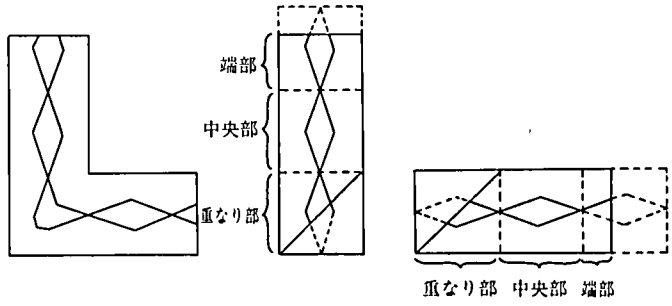


図12　罫線の生成手順

2.3.2 自 動 配 列

自動配列システムのポイントは二つある．一つは配列作業は確実に一定時間内に完了することである．本システムは新聞制作ラインの一環で当然のことである．しかし，自動配列は必ずしも正解を保証していない．たとえば，100行×2段の案内領域に対して60行物×3件の案内原稿は総面積比較では余裕があるが，原稿の折り返し（分断）が禁止されている制約のもとでは絶対に配列は不可能である．配列不可の場合には不可理由を明示することが重要である．案内領域の絶対量の不足，ある一定量以上の余白の発生は案内領域の変更を余儀なくされ，広告局だけでなく編集局を含む対応を迫られる．

あと一つは見栄えの問題である．新聞の求人欄を見るといくつかの配列のルールが読み取れる．原則として業種順・行数順・見出し順に配列されているが，よく見ると上段の最左端と次段の最右端で原則が守られていないケースもある．

ここで自動配列に関連する基本用語の解説をする．

- 業種　：代理店入力時に原稿属性として入力する職種コード
- 行数　：案内原稿の大きさを表す
- 見出し　：案内原稿の見た目の先頭3文字の文字コードと文字サイズ
  ：原稿作成はどの部分から開始しても構わないので，右上端に隣接する3文字を自動抽出する．
- 旗　：最上段の右端もしくは中央に配置される新聞社のロゴマーク
- 柱　：業種の最初の原稿の直前に配置されるカット
- 仕切罫　：原稿間を区切る罫線
- 流し順　：複数の案内領域の使用順番を規定する人間系の指示
- 調整原稿：余白を穴埋めするための無料原稿．社告・料金表等がある．
- 甘さ　：余裕（原稿量と領域量の差）が大きすぎる度合い

・辛さ　　：余裕が小さすぎる度合い

自動配列の操作手順を図13に示す．この手順で人間系で指示するのは，配列対象の選択・変動的配列条件の入力・結果の検証である．

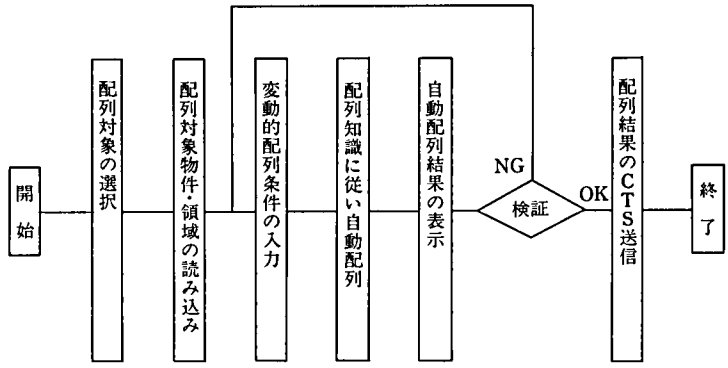


図13　自動配列システムの操作手順

1)　配列対象の選択……掲載日を指示する．自動配列システムは媒体支援システム・広告割付システムに要求を出し，該当日の案内対象物件と案内領域を入手する．媒体支援システム上に構築されたデータベース(ORACLE*を使用)に対する検索条件の設定により，対象物件は入手時には業種・行数・見出し順にソート済である．

2)　変動的配列条件の入力……配列条件には実行時に人間系の判断による変動的な条件と知識データベースに登録されている普遍的な条件がある．変動的条件の代表例は流し順と調整原稿の選択である．流し順は必須指示項目であるが，調整原稿の選択は一般に結果の検証の対処として利用する．

3)　結果の検証……検証作業は図6の表示を人間の目チェックで確認する．この時点では原稿内容は無関係であり，配列のバランスだけに注目する．NGと判断した場合には，通常は変動的配列条件を変更して再実行する．条件変更では処置不能と判断した場合には，案内領域の変更をすべく対外（編集局）調整を図る．広告割付システムでの案内領域の変更後，再度，配列対象の選択から処理を再開する．

配列条件が確定すると自動配列処理を開始する．エキスパートシステム(ES)の開発モデルとして，案内広告自動配列システムは計画・設計型の割当問題に分類される．本システムで採用のGNOSIS-IIは組み込み型のES構築支援ツールであり，強力な前向き推論エンジンを備えていることから，本開発モデルである計画・設計型の問題解決に威力を発揮する．

GNOSIS-IIでは専門家の持つ知識を，①フレームと関係，②プロダクション・ルール，③外部手続きとして表現する．

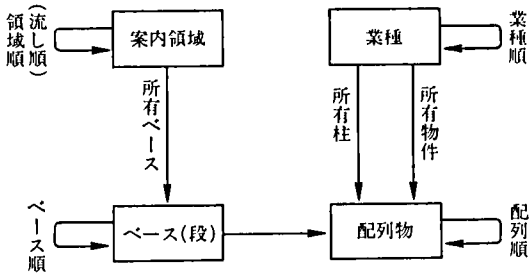
フレームは物・事象・状態といった事柄を統一的な概念でくくり，構造化したものである．関係は二つのフレーム間の結びつきであり，フレーム間に存在する階層関係や因果関係を表現する．いずれも宣言的な知識の表現に使用する．

* ORACLEは米国ORACLE社の登録商標である．

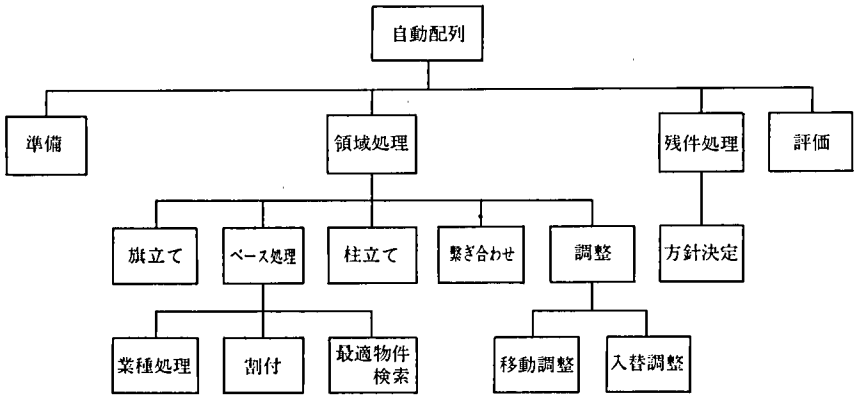
プロダクション・ルールは知識を「IF　条件　THEN　結論」という形で表現し，条件が成立すれば結論に指定した処理を実施するものである．

外部手続きは算術演算のように，ある結果や結論を導くために実施する一連の手順に関する知識を表現する．副プログラムの形式で作成し，GNOSIS-IIに埋め込むことにより利用する[3]．

本システムで定義しているフレーム数は10，関係数は29，プロダクション・ルール数は107である．知識ベースの構造を図14に示す．図の (a) フレーム/関係図では，フレーム型は枠内にフレーム名称を明記した長方形で，関係型は線上に関係名称を明記した長方形を結ぶ矢印付実線で表記している．図の (b) ルールモジュール構造図では，ルールモジュールを長方形で，名称を枠内に，モジュール間の関係を実線で表記している．



(a)　フレーム/関係図



(b)　ルールモジュール構造図

図14　自動配列システムの知識ベース

問題を整理すると，本システムは与えられた案内領域（実際はベース）に与えられた配列対象物（案内物件・旗・柱・仕切罫）をルールに従って配置する．このルールは見栄えよく配置するための条件であり，基本ルールと調整ルールがある．

1)　基本ルール

・業種順・行数順・見出し順に配置する．

・旗・柱を所定位置に立てる．

・物件間に指定の仕切罫を引く．
・各段の余裕量はなるべく一様になるよう物件間に割り振る．

2) 調整ルール

調整ルールは，基本ルールだけでは配列が未完の時に実施される．調整ルールは基本ルールを破るためのルールである．

・段の繋ぎ目で同一業種内の行数順ルールを破り，物件を入れ換える．
・業種順ルールを破り，異業種の物件を全件一括で入れ換える．
・物件間の仕切罫を細罫に切り替える．
・旗・柱を一部削除する．

上記ルールで最も実現が困難なルールは余裕量の一様割り振りである．シミュレーション・システムといえども本質的には順処理であり，本システムでも下（最後段）から上（最前段）へ順に配列を進める．固定余裕量の限界値を越える配置時に該当段の処理を完了すると，どうしても下方は辛い配置になり，上方は甘い配置となる．これは設計時（バブル期）の社会的背景もあり，あふれる広告を如何に掲載するかという辛さ調整に主眼が置かれていたことにもよる．

対応策として理想余裕の概念を導入した．理想余裕とは各段での余裕量を何行以下にするかの基準値である．計算式を下記に示す．

理想余裕＝（1－未配列物件の総行数/未配列領域の総行数）
×対象段の有効行×余裕系数

余裕係数は初期値1.00，最大値1.50，最小値0.00，増分0.25，減分0.50である．

理想余裕は準備Mで初期設定し，割付Mで1件の配置ごとに残余裕量≦理想余裕となるよう検査し，条件を満たせば該当段への割付を終了する．次の対象段を検索するベース処理Mでは理想余裕を再計算し，割付Mを繰り返し実行する．余裕係数の変更は割付M・ベース処理Mでは実施せず，シミュレーションの1サイクルの完了時に実行される方針決定M・評価Mにおいて結果を判断して増減され，自動配列は初期状態から再実行される(xxMの表記は図14のルールモジュール名称を表す)．この処置により，実用に耐え得る満足のいく成果が得られている．

3. システムの評価

システムの稼働開始から1年を経過した現時点（1994年2月）において，2章で設定した目標に対しての達成度は100%である．ただし，本番稼働時期（'94.2）が広告不況に遭遇し，本システムの導入は媒体（媒体システム・自動配列システム）と対象代理店（代理店システム）5社のうち，3社であった．しかし，導入を見合わせた2社も今年度中本番稼働（うち1社は，'94年6月本番稼働）が決定している．

システム導入の効果をコスト面，サービス面，ミス防止の観点で評価する．

1) 省コスト（省力化・省人化）

・代理店入力，自動配列の実現による大幅な外部委託費の削減
・原稿の代理店直接入力および配列の自動化により，校閲要員については2～3名，入力については94%削減(非定期的な企画物以外)，また媒体制作要員は1～2名程度削減されている．

2)　サービス向上

・自動配列の所要時間は目標20分に対して，通常5分で完了する．原稿の締切時間はすでに1時間延長され，更なる延長も検討されている．

・緊急時（編集側での突発ニュースの発生）にも時間的に対応が可能となる．

・本稿では触れていないが，代理店の広告主に対する顧客管理・媒体の代理店に対する顧客管理の充実が達成された．

3)　ミス防止

・当然のことながら，媒体での入力の廃止・ライフサイクルの自動管理により，掲載ミスは皆無である．

今後の課題としては，図7で明らかなような代理店営業マンは本システムの恩恵を享受していない．提案時点では数年後に携帯用UNIX*機の実現を予測して，デスクトップ型の本システムの入力機能を移植することを想定したが，現時点では重量面で期待に達していない．

4. おわりに

本システムはコスト面での絶大なる効果が期待でき，日本新聞協会の今後の方向性とも合致することから業界での関心は高く，日刊スポーツ新聞社の協力を得て弊社として商品化を決定した．商品化は，①携帯端末としてPCを想定して機能を移植，②サーバ側システムのSolaris 2への対応，③文字の標準化（著作権の関係上，現行文字の使用不可）を予定しており，PC移植から開発に着手している．配列のルール等は新聞社ごとに異なり，カスタマイズを必要とするが，本システムが幅広く利用・活用されることを期待する．最後に我々と一緒に開発に携わり，貴重な資料をご提供頂いた日刊スポーツ新聞社の関係諸氏に心から感謝の意を表したい．

---

* UNIXオペレーティングシステムは，UNIX System Laborations, Inc. が開発し，ライセンスしている・

参考文献　[1]「特集　広告のオンライン化」，新聞技術 No. 144, 1993.
[2]　東野康臣，「エキスパートシステムの開発方法とツール」，ユニシス技報 Vol. 13 No. 2, 1993. 8.
[3]「エキスパート・システム構築ツール GNOSIS-II解説書」，ユニシス・マニュアル，481745901.

執筆者紹介　近　藤　千　秋（Chiaki Kondo）
昭和49年東京教育大学理学部数学科卒業．同年日本ユニシス（株）入社．主に新聞社向けシステムの開発，保守を担当．現在，社会公共システム第2本部 システム5部第1課課長．

A6-NSによる高信頼性システムの構築

The Creation of a Highly Reliable System Based on the A6-NS

小　池　　卓

要　約　今日，流通業界においては，競合他社との差別化を図るための顧客サービス競争の結果，システム規模の拡大や長時間連続運転への期待が大きくなっている．情報システム部門では，障害の発生に対して，エンドユーザのサービスを停止しないようなシステムを実現すべく努力している．

本稿では，総合スーパーのO社でのA6-NSシステムの導入事例を通して，その分割運用技術の利用と，「耐障害性」を高めたシステム構成を構築経緯を含め紹介する．

Abstract　Businesses in today's distribution industry find themselves competing with each other for better customer services in an attempt to have their own specialized lines recognized by customers, thus boosting computer systems needs for further expansion in scale and non-stop operation for a longer time. Then, information systems departments have been forced to struggle to build such a system as guarantee uninterrupted end-user services even when any systems failure happens.

By picking out a sample case of the A6-NS mainframe installed at the O company, a general supermarket chain operator, the author describes how its split data processing technique works there and how the system with enhanced 'fault tolerance' has been created.

1. はじめに

コンピュータが一般消費者と直結し，その障害が社会問題にすら発展するような，金融業界や公共事業，航空業界などにおいて，早くから無停止システムなど信頼性対策に対する投資が行われてきた．今日，流通業界においても，『商品の多品種，小ロット，短サイクル化』，『年中無休，24時間営業』など，消費者ニーズの多様化，高度化への対応を迫られている．これにより年々，システム規模の拡大や長時間連続運転の必要性が大きくなっている．

総合スーパーのO社は，これまで単一プロセッサのA6-FSシステムおよびA3-Eシステムのユーザであった．O社では1991年度よりシステム化長期計画が策定され，システム整備が進められてきた．この中で『衣料システム』の新規開発，および生鮮食料品の加工配送センタ（以下PCセンタ）の開設に備えた『生鮮システム』の新規開発が経営方針として掲げられた．それにともないコンピュータ能力のレベルアップと24時間安定稼働を目標にしたシステム構成の選定を行った．

O社のシステム構想と設備投資予算の中で，どのようなシステム構成を組めば『信頼性の高いシステム』が構築できるかを考察した経緯を以下に述べる．これまでにも，AシリーズのA-CAF*（A series Continuous Application Facility）のような本番系ホストから待機系ホストに自動的に切り換える『ホットスタンバイ・システム』や，

*A-CAF：自系内監視制御：ホットスタンバイシステムを核としたオンライントランザクション処理連続運転支援システム．対象機種はA11以上のシステム．

2200シリーズのXTPA*（eXtended Transaction Processing Architecture）などを利用した大規模な『無停止システム』は存在した．しかし，一般的に無停止システムはハードウェアの冗長性の上に成り立っており，投資規模から見ても，中小型汎用機のマーケットにおける無停止システムの実現はほとんどなかった[1]．本稿はAシリーズの中型機であるA6-NSシステム（密結合4プロセッサ機）を中心として構成した『耐障害性の高いシステム』の導入事例報告である．A6-NSシステムの分割運用機能を利用して，ハードウェアの障害に対し最短時間で処理が回復できるよう構成されたシステム構築の事例である．これは『無停止システム』とは異なるが，中小規模システムの安全性を高めた好例と考え，ここに紹介する．

2. コンピュータ障害対策の現状

2.1 システムダウンの状況

(財)日本情報処理開発協会が1991年7月に実施した「情報セキュリティに関する調査」に基づいた情報化白書（1992年度版）[2]では，以下のように言われている．

「過去1年間にシステムダウンを経験したユーザは，全体の58.6%と過半数を超え

表1　過去1年間(1990年7月～91年6月)のシステムダウン発生件数[2]

1	発生した	1,015件	58.6%
2	発生しない	691	39.9
	無回答	25	1.4
	計	1,731	100.0

表2　原因別，平均システムダウン発生率(多重回答)[2]

原　因	回答数	延べ回数/987
自然災害	126	0.21回
電源障害	290	0.42
空調等障害	124	0.17
回線障害	208	0.89
ハード，OS障害	611	1.73
ソフト障害	323	1.11
火災による事故，障害	1	0.00
人の悪意による事故等	2	0.01
オペミス等 人の過失による事故等	232	0.64
その他	27	0.09
合　計	987	5.25

(注)これは，過去1年間にシステムダウンを経験した1,015社のうち，障害原因別に回答企業の延べ回数を，当該項目に回答した987社で除したものであり，987社における1社当たりの原因別障害発生可能性を示す．

システムダウンとは，システムの全面ストップもしくはそれに準じる障害と定義する．1回の事故について原因が複数考えられる場合は主要原因のみ記入．

*XTPA：フォールトトレラントなレコードロック・プロセッサが複数ホスト間でのレコードロック制御を行う疎結合マルチホスト・システム．

ている(表1)．また，システムダウン発生の原因は，『ハードウェア，OS障害』，『ソフトウェア障害』が多い(表2)．平均ダウン間隔時間(MTBD：Mean Time Between Downs)と平均ダウン時間(MDT：Mean Downs Time)を見ると，MTBDは過去3年間で3.27倍と大幅に伸びており，システムダウンが減少して安定していることがわかる．しかし，MDTも過去3年間で1.72倍と長くなっていることにも注目しなければならない．すなわち，システムダウンは減少傾向にあるが，発生した場合には従来よりも長時間停止していることを示している(表3)」．

表3　MTBDとMDT[2]

	今回調査	前回調査
MTBD	3,572時間	1,090時間
MDT	101.7分	59.1分

(注) 前回調査は1988年に実施

2.2　障害対策の状況

障害対策として最も一般的に行われているのは，データやプログラムをコピーして他の場所に保管したり，緊急時の手作業による修復手順をマニュアルとして整備するなどの『バックアップ体制』である(表4)．年々システムが複雑化しMDTが長くなっていることを考えると，MDTをできるだけ短くするための障害対策も必要になる．そのためには，『CPUのデュアル・システム化』や『回線の二重化』などが有効である．しかし，二重化による障害対策は一般に冗長な設備投資につながるケースが多く，全産業平均で10数%とあまり浸透していないようである．

表4　信頼性対策(業種別)[2]抜粋

信頼性対策 / 業種別		回答実数	自己診断システム保有	定期診断システム制	バックアップ体制	回線の二重化	CPUデュアルシステム
全産業計		1.007	397	491	731	136	154
		100.0	39.4	48.8	72.6	13.5	15.3
主な業種	小売業	33	14	15	23	5	3
		100.0	42.4	45.5	69.7	15.2	9.1
	金融業	97	53	61	81	41	59
		100.0	54.6	62.9	83.5	42.3	60.8
	電力・ガス事業	10	4	4	7	2	2
		100.0	40.0	40.0	70.0	20.0	20.0

(各業種ごとに上段：社数，下段：%)

2.3　Aシリーズの障害対策の現状

Aシリーズは，システム障害を事前に検出する「自己診断システム」として，独立した保守診断プロセッサを保有している．また，技術員による「定期診断」も保守サービス契約の一環として行われている．「バックアップ体制」は，顧客責任のもとほとんどのユーザで実施されていると考えている．「回線やCPUの二重化」等は，金融関係や大手企業を中心に多く導入されているが，それ以外の中小規模システムでは多く

ない.

3. O社の概要

3.1 企業概要

O社は，衣料，生鮮，日配品を扱った総合スーパーマーケット・チェーンをY県内に展開している．大型店で県下四方を固め，その間をミニスーパーで補う出店計画により1994年1月現在の店舗数は25店舗を数え，1997年には50店舗，売上1千億円達成を目指している．

3.2 システム概要

O社では1991年度からシステム化長期計画の中で，店舗，取引先とのオンライン化(オンライン受発注システム；EOS: Electronic Ordering System)，POSシステムの導入などを推進し，商品管理，売上管理，特売管理，利益管理，財務会計，一般会計，給与計算などの業務をシステム化してきた．さらに5年後の売上目標を実現するため，事業展開のインフラとしての「経営情報総合システム」の開発を目指し，システム分析を開始した．まず新システムの足がかりとして，今回の「衣料システム」および「生鮮システム」の開発と，24時間365日安定稼働を可能とするシステムの構築を目指している．

システム開発は，第四世代言語LINCによりオンライン・データベース・システムを構築し，帳表類は独自のプログラミングの標準化を基にCOBOLで進められている．O社システム担当者より，LINCの生産性は高く評価されている．

これまで，基幹業務をA6-FSシステムで行い，開発・テストをA3-Eシステム，「財務システム」をU5000/85システムで実行している．また，POSシステムのFEP(Front End Processor)は他社機UNIX*ベースマシンで実行している．その他，カットシート・レーザービーム・プリンタをオフラインで使用している．

それぞれのホストで独立したネットワークを持っており，A6-FSとA3-Eの間はLAN接続され，A6-FSとU5000/85の間をPOL/SEL(Aシリーズ独自の通信プロトコル)で接続してデータ交換を行っている．FEPやレーザビーム・プリンタとのデータ交換は，磁気テープを媒体にしている．

3.3 障害対策の現状

障害対策としては，Aシリーズ・ユーザに標準に提供されている「自己診断システム」および「定期診断システム制」と，主要業務の「EOS」に関連するデータベースを定期的にバックアップすることで対応している．この業務は，各店舗のEOS端末から発注情報を随時公衆回線6回線(JCA手順)で集信し，商品マスタに突き合わせて集約し，このデータを毎日数回，VAN業者に対し交換回線(DDX-C/9600 bps)で送り出すオンライン受発注業務である．O社はA6-FSとA3-E 2台のAシリーズ・ホストを有しており，A6-FSダウン時に店舗からの発注データをA3-Eで代行受信することができる．しかし，商品マスタを両システムで同レベルに保つことができないためEOSの継続稼働は機能せず，優先度の高いものから電話やFAX発注で対応しているのが現状である．

*UNIXオペレーティング・システムは，UNIX System Laboratories, Inc.が開発し，ライセンスしている．

4. 提案システム構成決定の背景

4.1 システム構成への要望点

システム構成を決定するにあたってのO社の要望事項は以下のとおりである.

1) 『EOS』業務の無停止……『EOS』業務が停止するとVAN業者から卸売り業者へのデータ配信ができず,翌日の店舗の品ぞろえに影響するため,この業務の続行を最優先に考えること.

2) 『生鮮システム』の無停止……システム・ダウンが発生すると24時間稼働のPCセンタでは値付け機による加工指示が停止する.これは即,生鮮品の店舗への供給に影響を与える上に,加工作業員の手も止まる.このため最悪の場合でも代替手段をとれるようにすること.

3) 設備投資の制限……設備を単に二重化するのではなく,冗長性の少ない構成の中で障害対策を実現する.また,これまでのシステム資産を極力利用すること.

4) 障害の局所化……障害箇所を確定できたら,それを他に波及させないシステムであること.

5) システムの拡張性……データの増大や,今後のシステム開発に耐え得るシステムであること.またハードウェアの追加や,上位機種への拡張性があること.

6) ネットワークの統合……それぞれのホストシステムが独立したネットワークを持っているため,これを統合すること.

4.2 ホストコンピュータの決定

ホストコンピュータは,既存のシステム資産がある上,今後もLINCによる基幹システムの開発を行うことで合意を得ており,Aシリーズのアップグレードを基本に考えた.システム規模から要求されるプロセッサ能力は,少なくともA6-NS以上と見られた.当時の製品ラインアップから,提案機種はA6-NSあるいはA16以上の機種が候補に上がった.A6-NSは密結合4プロセッサ・システムで,A16のシングルプロセッサ・システムより安価に構築できる.設備投資の制限からA16のマルチプロセッサ・システムで構成を組むことは困難である.このため,経済性,安全性,信頼性,効率性などから検討し,A6-NSを中心にして,O社の要望事項をいかに包含したシステム構成にするかを重点に検討した.

なお,これ以降A6-NS(中小型機;EMS: Entry Micro System)用の用語で説明する.A4/A6のFS/HS/KS/NS(モデルS)は,プロセッサにSCAMP(Single Chip A series Mainframe Processor)チップを採用し,プロセッサを1チップ化したコンパクトなシステムを実現した.旧来のA1/A4/A6システムをSPARTANシリーズと称したのに対し,モデルSをGEMINIシリーズと称している.A6-KSおよびA6-NSは,分割運用(5.2節参照)が可能な構成を組めるマルチプロセッサ・システムである.

5. A6-NSシステムの耐障害性機能

5.1 基本システム構成

Aシリーズが独自のオペレーティング・システム(MCP/AS: Master Control Program/A Series,以下MCPと略す)の下で走行するための最小限必要な構成(MCPが制御・管理しているメインフレーム構成を「グループ」と呼ぶ)要素は次の

とおりである．

1) データ・プロセッサ……MCP のコピーを使用して，適用業務プログラムとシステム・プログラムを実行する．中央処理装置（CPU：Central Processing Unit）が EMS におけるデータ・プロセッサである．

2) 入出力プロセッサ……データ・プロセッサのために，メモリと入出力（I/O）サブシステムとの間でデータを転送する．GEMINI シリーズでは，入出力アダプタ（IOA：Input Output Adapter）という．

3) 入出力サブシステム……MCP と周辺装置との間のデータ転送を管理する一連の特殊なプロセッサである．構成要素には，入出力(I/O)ベース*，DLP**(Data Link Processor)，周辺装置を含む．

4) メモリ・サブシステム……メモリ制御装置と記憶装置からなり，主メモリとデータ・プロセッサの間のデータ転送をすべて扱う．

5) システム制御サブシステム（SCS：System Control Subsystem）……EMS において保守機能を制御するハードウェアおよびソフトウェアである．

A6-NS は，最低二つの基本キャビネットで構成される．それぞれのキャビネットにデータ・プロセッサ，I/O プロセッサ（以下 IOA），メモリ・サブシステム，内蔵の I/O ベースと DLP，および内蔵ディスクが格納される．この機種は A6-KS と共に EMS において，はじめてシステムの分割運用が可能になったモデルである．分割運用において，システムは，最低一つの基本キャビネットを持たねばならない（図 1）．

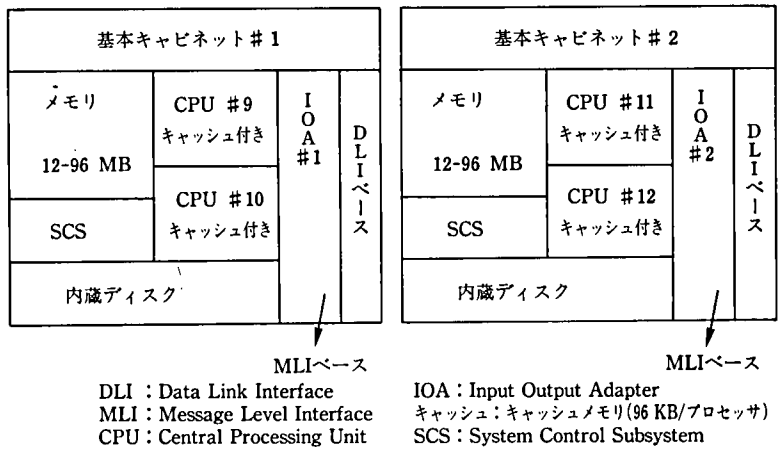


図 1　A6-NS ブロックダイアグラム

A6-NS は周辺装置の数に従い，最高四つの拡張キャビネットの接続が可能である．拡張キャビネットには I/O ベースと DLP，内蔵ディスクなどが格納される．

基本キャビネット内の I/O ベースは，DLI（Data Link Interface）と呼ぶインタフェースで接続されるので「DLI ベース」と言うこともある，拡張キャビネット内の I/O ベースは，MLI（Message Level Interface）と呼ぶインタフェースで接続されるの

*入出力（I/O）ベース：IOA を経由して，ホストのデータプロセッサと DLP との間でメッセージの経路選択を制御する．
**DLP：ディスクやテープなど特定の型ごとに周辺装置を制御するハードウェアで，「チャネルプログラム」を内蔵する．

で「MLIベース」とも言う.

5.2 A6-NSシステムにおける分割運用

図1の二つのキャビネットは，5.1節の1）から5）の基本システム構成の条件を満たしているため，それぞれ独立したグループとして分割運用できる．A6-NSは4プロセッサ・システムであるが，分割できる単位は2プロセッサずつ二つのグループになる．ただし，それぞれのキャビネットに接続されたディスク装置の中に，MCPが登録されている必要がある．

O社におけるA6-NSの運用は，必要なプロセッサの処理能力から分割運用を前提としていないが，一部のハードウェアの障害に対し，処理能力の低下があっても，業務処理を続行させることが要求されている．そのため，障害のあるハードウェアを切り離しても，必要なデータをアクセスできる経路が存在するようなハードウェア構成が必要になる．分割運用におけるハードウェア構成は，両システムから，必要な時に(たとえば，一方のシステムの障害に対し，他方のシステムから)必要なデータのアクセスを可能にする構成をどう組むかが重要な問題である．この問題は，非分割運用における耐障害性の向上のためのハードウェア構成と同じ問題であり，したがって分割運用でのハードウェア構成の組み方から検討する．

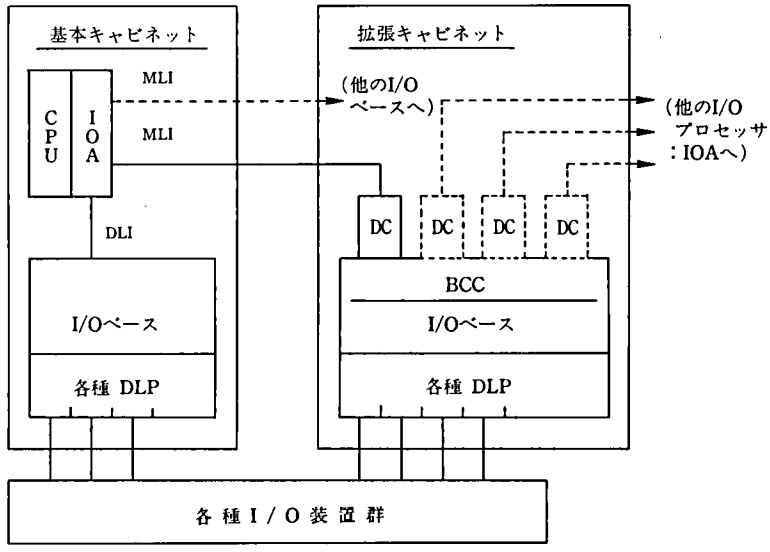
分割運用の形態の一つとして，それぞれのグループを別々のコンピュータとして稼働させることが考えられる．この場合それぞれのMCPのバージョンは同一である必要はない．たとえば，開発によるプロセッサ負荷が主要業務に影響しないようにプロセッサやメモリを明確に分割する場合や，新旧システムの並行処理などに利用できる．

もう一つの形態として非分割運用システムの障害対策に利用することが考えられる．A6-NS/KSの場合，非分割運用でプロセッサ障害時にプロセッサの自動切り離しによる処理の継続はできない．分割運用機能を利用して，障害のあるプロセッサを手動で切りはなすことで障害の再発を防止する．このとき構成変更を伴ったホルトロード（システム初期設定）が必要である．

プロセッサ障害時，そのキャビネットのIOAから接続された周辺装置は使用できない．周辺装置をプロセッサ障害の影響から避けるためには，障害のないキャビネットのIOAからも周辺装置との経路を設け二重化しておく必要がある．

5.3 I/Oベース・シェアリングによる耐障害性

分割運用においてプロセッサ障害が発生した時に，正常なプロセッサに接続されたIOAから経路づけされたI/Oベースは障害の影響を受けない．複数のIOAと経路づけられたI/Oベースは，プロセッサ障害の影響を回避できる．IOAはDLI1本とMLI2本の接続が可能である．DLIはIOAと基本キャビネット内のI/Oベースを接続する場合に用いられるインタフェースである．このI/Oベースは，複数のIOAと共有できない．MLIはIOAと拡張キャビネット内のI/Oベースを接続する場合に用い，このI/Oベースは最大四つのIOAから共有される．共有の時，I/OベースへのIOAからの接続は，DC（Distribution Card）を介して行い，共有時におけるIOAからのアクセス許可/不許可を示すビットマップを持つBCC(Base Control Card)が必要となる(図2)．複数のIOAで共有されたI/Oベースの接続をI/Oベース・シェアリングと呼び，プロセッサ，IOA等の基本キャビネット内の障害に対し，周辺機器への入出力経路を



IOA : Input Output Adapter　　DC : Distribution Card
MLI : Message Level Interface　　BCC : Base Control Card
DLI : Data Link Interface　　DLP : Data Link Processor

図 2　IOA と I/O ベースの接続

維持できる.

5.4　A6-NS のディスク装置の接続

周辺機器の接続を代表してディスク装置を例に説明する. ディスクは大別して内蔵ディスクと外部ディスクの2種類がある. 接続方法には, 同一キャビネットに存在するI/Oベース内の一つまたは二つのDLPから接続される場合や, 他キャビネットに存在するI/Oベース内のDLPからのパスと共に接続される場合などがある. 内蔵ディスクは, パスを一つしか持っておらず, 接続法としては図3の①, ⑤の2通りが考えられる. 外部ディスクの場合, パスを二つ持つデュアル・ポート接続ができるものもあるため①～⑥の接続方法がある. デュアル・ポート接続は二つのDLPから周辺装置への経路を二重化できるため, DLPの入出力負荷分散と共に耐障害性を高める効果を持つ.

BOOTユニット*として使用する場合はDLI接続されている必要があり, ①, ②, ③, ⑥の接続形態になる. ただし, A6-NSなどで, 分割運用する場合は, PC-ODT(ワークステーションPW²ファミリをコンソールにする) にBOOTが存在しているため, 「DLI接続されている」と言う制限はない. 分割運用におけるそれぞれの接続方法につき, 障害箇所をCPU, IOA, I/Oベース, DLP, パスと仮定した場合のそれぞれのディスク装置の可視性と特徴を表5に示す.

5.5　ディスク装置への耐障害性の高い接続方法

非分割運用において, 一方のキャビネットのCPUやIOAに障害が発生した場合,

*BOOTユニット：AシリーズがMCPのもとで稼働するための初期動作を行うマイクロ・コード・ファイル (BOOT CODE FILE) の存在するディスク装置.

表5　ディスク装置の接続方法と特徴

○：使用可能　X：使用不可
△：条件により使用可能(注3)

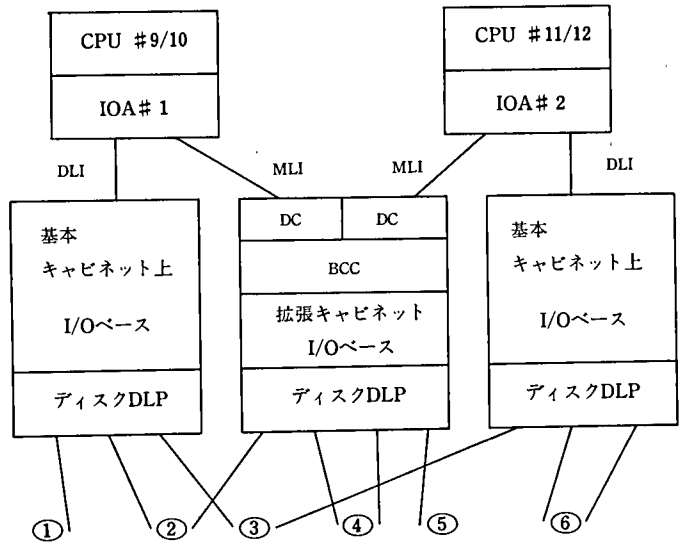
障害箇所＼ディスク装置	①	②	③	④	⑤	⑥	備　考
CPU#9/10 IOA#1	X	O	O	O	O	O	②④⑤のMLI接続のDLPがIOA#1経由ならDLP単位のACQUIRE*が必要(注1)，③のディスクはユニット単位のACQUIRE*が必要なケースあり(注2)．
CPU#11/12 IOA#2	O	O	O	O	O	X	②④⑤のMLI接続のDLPがIOA#2経由ならDLP単位のACQUIRE*が必要(注1)，③のディスクはユニット単位のACQUIRE*が必要なケースあり(注2)．
ディスク装置をサービスしているI/Oベース　一台	X	O	O	△ 注3	X	X	②③のディスクはユニット単位のACQUIRE*が必要なケースあり(注2)．
ディスク装置をサービスしているディスクDLP一台	X	O	O	O	X	O	複数パスを持つディスクはユニット単位のACQUIRE*が必要なケースあり(注2)．
パス	X	O	O	O	X	O	複数パスを持つディスクはユニット単位のACQUIRE*が必要なケースあり(注2)．

(注1) CPU障害の場合，基本的に再構成を伴ったホルトロードが必要．

(注2) GEMINIの場合，MCPオプションのPATHBALANCING*は使用できず，それぞれのDLPにユニット番号の小さいものから順に交互に割り付けられる，このDLPがどのIOAを経由しているかにより，装置を獲得する必要がでる．

(注3) 同一I/Oベースに二つのDLPを格納している場合は'X'で，DLPが別々のI/Oベースに格納しているときは'O'，拡張キャビネット二つはI/Oベースを搭載できるタイプもある．

*ACQUIRE：Aシリーズの ODT コマンドで装置，DLPなど資源を稼働中の「グループ」に獲得する操作を行う．
**PATHBALANCING：AシリーズのMCPオプション，EMSにおいてパスを動的に平衡化する．



(注)I/Oベース上のDLPの数はDLPの種類ごとに制限があるが図中では無視する．

図3　ディスク装置の接続

再構成（ホルトロード）が必要であるが，I/Oベース・シェアリングしていれば，装置の可視性を保つことができる．また，障害が発生していないキャビネットが一つのグループとして独立した構成（分割運用で稼働している）であれば，このグループに接続された装置は影響されない．つまり，I/OベースシェアリングされたMLIベースに接続する④，⑤や，異なるIOAの下にあるI/Oベースにデュアル・ポートで接続する②，③が耐障害性が高い．とくに②，③については，キャビネットの障害に対しても耐障害性がある．適用業務の処理を継続するために必要なハードウェアはディスク装置に限らず，これらの耐障害性の高い接続をする方がよい．

ディスク装置自体の障害をシステムの稼働に影響させないためには，Aシリーズのミラー・ディスク機能を使用する．これは，あるディスク装置とまったく同一の内容の複数のディスク装置を生成する機能である．ディスクの二重化をはかり，一つのディスクに障害が発生した場合に，他のミラー化されたディスクによって処理を継続することができる．

5.6 A6-NSのネットワークにおける耐障害性

ネットワークの耐障害性を考えるとき，通信制御装置に関するものとモデムなどを含めた通信回線に関するものの双方の障害の可能性を考慮する必要がある．現在，Aシリーズは，EDC-DLP(拡張データコムDLP)をはじめとするDLPタイプのものと，多機能通信プロセッサであるCP2000のフロントエンドタイプのものの2種類の通信制御装置をもつ．

EDC-DLPは，接続される回線および端末に対して，同時に経路を二重化できるプロトコルを搭載していないこともあり，図3のディスクDLPをEDC-DLPに置き換えると，端末は①，⑤のいずれかの接続形態になる．この場合の耐障害性は表5に準ずる．EDC-DLP自体の障害を想定すると予備のDLPを待機させる以外手段がなく，回線の移動は物理的に繋ぎ替えるか，または回線スウィッチング・ユニットを使用することになる．

CP2000は，AシリーズとLAN経由で接続する．この時Aシリーズ側にはICPというDLP型のハードウェアが必要であり，この間はCPLANというIEEE802.3準拠のプロトコルで接続される．この間のメッセージはダイナミックにルーティングされるためICP, LANケーブル，CP2000のLANインタフェース・カード共に多重化が可能である．図3のディスクDLPをICPに置き換えて見ると，CP2000の接続は①～⑥のすべての形態が可能である．CP2000は接続する電気的インタフェースやポートの数によってボードを選択し装填するタイプの機器であり，導入されたボード構成により障害対策の内容も異なってくる．一般的に耐障害性を高めるためには，構成要素を二重化することが必要になる．通信回線の耐障害性を考える場合も二重化あるいは代替経路を設定して対処する．

6. O社システム構成の選定

6.1 要望事項の検討

4章，5章で述べた内容をもとに，システム構成を選定した．4.1節で述べたO社の要望事項は大別して要約すると以下のとおりである．

① 耐障害性の高いシステム構成，すなわち重要業務の無停止
② 設備投資の制限
③ システムの拡張性
④ ネットワークの統合

ホスト・コンピュータは，さきに述べたようにA6-NSに決定した．これは一つには，新規開発のシステムを含めて要求されるであろうプロセッサ能力からA6-NS程度のホストコンピュータが想定されたためである．また一つには，A6-NSはO社がこれまで使用してきたA6-FSと同一筐体を使用しているため，無駄なくシステムのレベルアップができることが設備投資予算の面から見て大きな決定要因であった．基本的にA6-FSからA6-NSへのレベルアップでは，プロセッサのフィールド・アップグレードが可能である．そして最大の決定要因としてA6-NSが分割運用可能なマルチプロセッサ・システムであるため，この機能を利用してホストコンピュータ・システムの耐障害性を高められるところにある．障害のあるプロセッサ系を切り離したときプロセッサ能力は半減するが，ディスク等必要ハードウェアさえ使用できれば，最重要業務の実行に絞ることでO社の要望に応えることができる．

耐障害性を高めるために，拡張キャビネットはI/Oベース・シェアリングする形で接続した．また，ディスク装置はデュアル・ポート接続できるM9710型が採用された．これまでA6-FSで使用していたZD型ディスク装置は1パスのものであったため，A3-Eにつなぎ換えて開発用に転用された．

システムの拡張性については，ディスク，メモリとも追加の余地は十分あり，プロセッサ能力が間に合わない場合にもAシリーズ上位機種へユーザ・プログラムの修正なしにレベルアップすることができる．

ネットワークの統合の要素として，これまで財務処理で使用していたU5000/85とA6-FSの間が，POL/SELと言うAシリーズ独自のプロトコルでオンライン接続されており，これの標準プロトコル化や，POSシステムのFEP(UNIXベースのマシン)との間をオフラインで行っていたデータ交換処理のオンライン化という要望があった．また，別途U6000をホストとして『人事情報システム』をMAPPER-Cで開発検討中であり，ともにUNIXマシンであるため，この3台のネットワークをTCP/IPプロトコルで統合することにした．AシリーズはCP2000を介してTCP/IPに乗り入れられるため，これを導入してAシリーズ2台とUNIXマシン3台の間にTCP/IPネットワークが構築できた（図4）．

O社の『EOS』は，構造上CP2000環境およびCOMS（COmmunication Management System：Aシリーズ通信管理システム）対応プログラムを想定しておらず，プログラム修正の費用からEDC-DLP環境でこれまでどおり使用することにした．このため『EOS』は，EDC-DLPの障害時に動作しないことになってしまう．O社の『EOS』の無停止の要望に応えるためには，別の方法でのバックアップを考える必要がある．A3-EにもEDC-DLPと同等の機能を持つNSP/LSP(ネットワーク制御プロセッサ)があり，空き回線もあることから，これをバックアップに利用することにした．6.3節と6.4節で，バックアップの仕組みを説明する．

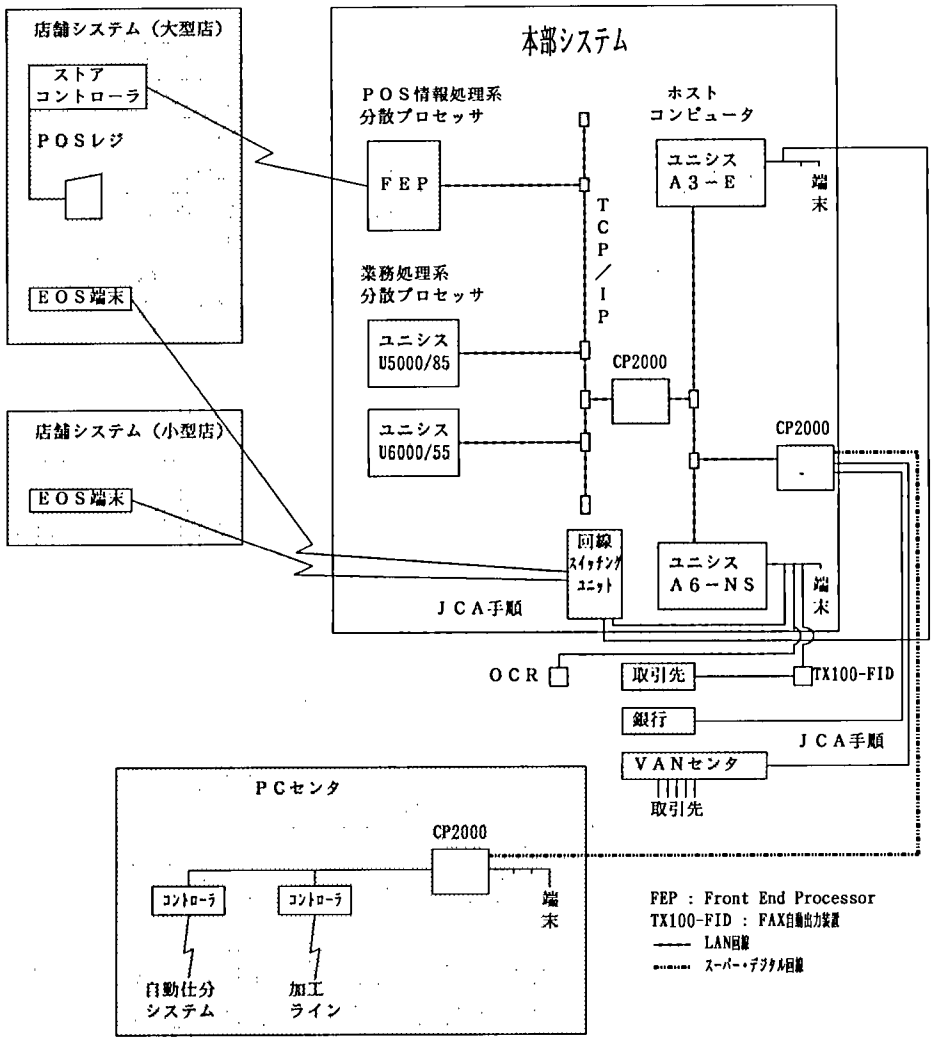


図 4　TCP/IP によるシステム間接続図

6.2　DLP 構成

接続する周辺装置の数から，基本キャビネット 2 台，拡張キャビネット 2 台，つまり DLI ベース x2，MLI ベース x2 の構成とした．MLI ベースは I/O ベース・シェアリングしているため，これに接続された DLP は，すでに IOA からの経路が二重化されており耐障害性が高い．したがって，重要度の高い DLP でユニットへの経路が二重化できないものを優先してここに配置する．DLI ベースには，ユニットへの経路が二重化できる装置を制御する DLP の一方や，比較的優先度の低い DLP などを配置する．システム全体でのディスク DLP の数，キャビネットごとの DLP の数，I/O ベースごとのディスク DLP の数など，DLP の配置を決定する上で制限となる項目があり，これに上記の要件を照合して構成を決定した（表 6）．

6.3　A6-NS 障害時のバックアップ方法

A6-NS の耐障害性を高めても，たとえば一度に両方のプロセッサ系に障害が発生

表6 DLP構成

	接続されたDLP	接続された周辺機器(装置番号)
基本キャビネット#1	SCSI 1, ODT 2, ICP 1	PK 52-55, SC 1-3, NP 100
基本キャビネット#2	SCSI 1, ODT 2	PK 60-63, SC 172-174
拡張キャビネット#1	SCSI 1×2, LBP, TP 3	PK 60-63, 76-79, LP 140, LP 21
拡張キャビネット#2	SCSI 1×2, NSP 5×4 MTFIPS 1, ICP 1	PK 52-55, 68-71, DC 112-115 MT 156-158, NP 101

SCSI 1：ディスクDLP (PK)　ODT 2：コンソールDLP (SC)
ICP 1 ：LAN用DLP (NP)　LBP ：レーザビームプリンタDLP (LP)
TP 3 ：ラインプリンタDLP (LP)　NSP 5：拡張データコムDLP (DC)
MTFIPS 1：高速テープDLP (MT)

表7 データ配置

ユニット番号	ディスク名	内　　容
PK 52(68)	A(MIRROR)	プログラムソース/コード，ワークフロー，LINC
PK 53(69)	B(MIRROR)	データベース・ファイル，非データベース・ファイル
PK 54(70)	C(MIRROR)	データベース・ファイル，非データベース・ファイル
PK 55(71)	DISK (MIRROR)	MCP/AS，システム・ソフトウェア，環境ソフトウェア各種
PK 60(76)	D(MIRROR)	データベース追跡記録ファイル，プリント・ファイル
PK 61(77)	E(MIRROR)	データベース・ファイル，非データベース・ファイル
PK 62	F	データベース・バックアップ用エリア，非常用MCP
PK 63	G	新規システム開発用
PK 78	H	新規システム開発用
PK 79	I	非データベース退避用エリア，非常用MCP

した場合は処理の継続は不可能である．開発機として使用するA3-EがA6-NS障害時のバックアップ機として使用できれば，『EOS』業務および『生鮮システム』の無停止の要望に条件付きではあるが対応できる．A6-NS全体，あるいはEDC-DLPが集中する拡張キャビネットのI/Oベースが使用不能と判断されたときに，いかに早く本業務データベースをA3-Eに渡すことができるかが課題となった．

A6-NSのMLIベースは他のAシリーズ・ホスト（A1/A4/A6/A9/A10）からもI/Oベース・シェアリングすることができるが，A3-Eはその対象外である．業務データベースのあるディスクをA3-Eからも獲得可能にするため，ディスクへのデュアル・ポート接続するパスの一方をA6-NS，もう一方をA3-Eに接続して，本業務中はA3-E側からアクセスしないことにした．また，これらのディスクのミラー・メンバー(ミラーディスクの相手方のユニット)をもう一筐体のM9710に設定することにより，A6-NS非分割運用時の安全性を高めた（図5,表7）．

6.4 ソフトウェア構成ファイル

EMSシステムにおいて，I/Oベース・シェアリングされたシステムでは，とくに指定しなければホルトロード時に先に起動したIOAが可視性のある範囲のすべてのDLPの制御を獲得してしまう．この場合，入出力が片側のIOAに集中して実行されるため，効率が著しく低下する．効率を高めるためにはDLPをバランスよく両方のIOA

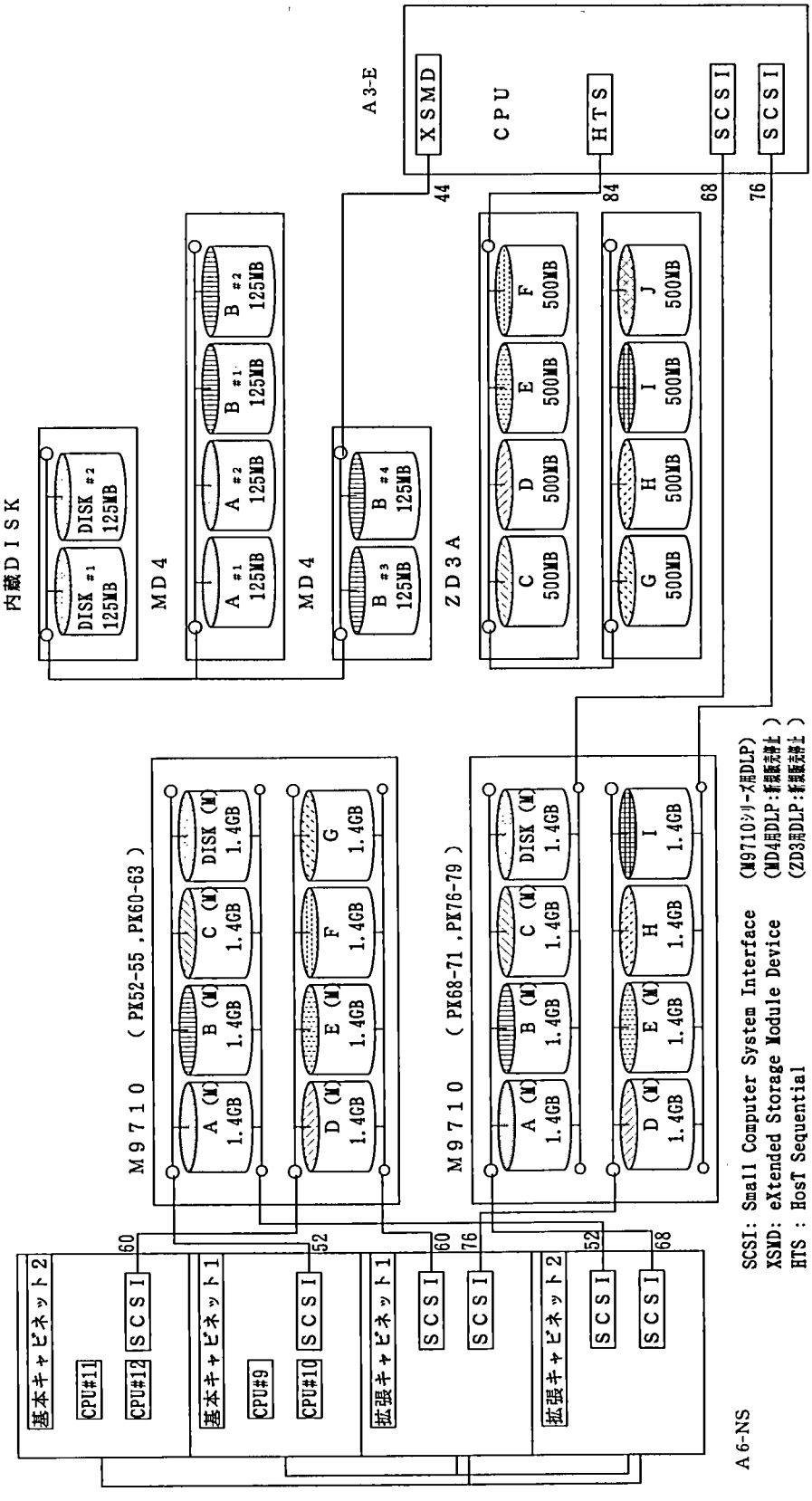

A6-NS
基本キャビネット2
CPU#11
CPU#12
SCSI
基本キャビネット1
CPU#9
CPU#10
SCSI
拡張キャビネット1
SCSI
SCSI
拡張キャビネット2
SCSI
SCSI
60
52
60
76
52
68
M9710 (PK52-55 , PK60-63)
A (M) 1.4GB
B (M) 1.4GB
C (M) 1.4GB
DISK (M) 1.4GB
D (M) 1.4GB
E (M) 1.4GB
F 1.4GB
G 1.4GB
M9710 (PK68-71 , PK76-79)
A (M) 1.4GB
B (M) 1.4GB
C (M) 1.4GB
DISK (M) 1.4GB
D (M) 1.4GB
E (M) 1.4GB
H 1.4GB
I 1.4GB
内蔵DISK
DISK #1 125MB
DISK #2 125MB
MD4
A #1 125MB
A #2 125MB
B #1 125MB
B #2 125MB
MD4
B #3 125MB
B #4 125MB
ZD3A
C 500MB
D 500MB
E 500MB
F 500MB
G 500MB
H 500MB
I 500MB
J 500MB
A3-E
XSMD
CPU
HTS
SCSI
SCSI
44
84
68
76
SCSI: Small Computer System Interface (M9710シリーズ用DLP)
XSMD: eXtended Storage Module Device (MD4用DLP:新規販売終止)
HTS : HosT Sequential (ZD3用DLP:新規販売終止)
(M) = Mirror disk GB = Giga Byte MB = Mega Byte

図5 ディスク構成

から使用できるようになってなければならない．また，CPU や IOA の障害を考慮して障害箇所を切り離した構成をシステムに登録しておくことも必要である．このため，I/O バランスやさまざまなトラブルを想定したソフトウェア構成ファイル (CONFIGURATION FILE) を作成する必要がある．

O 社では A6-NS 側だけでなく，A3-E の側にもソフトウェア構成ファイルを作成した．これは，A6-NS と A3-E から M9710 型ディスク装置に同時にパスを設定してディスクを共有したいためである．A3-E 側でソフトウェア構成ファイルを作成して，通常はこのディスクをアクセスしないよう定義した．また，本業務のホストを A3-E に切り換える時のための構成をあらかじめ次のように登録した．

1) ネットワーク (CP2000) のために A3-E のホスト名を A6-NS の名前に変更する．
2) A6-NS の持っていたデータを最終形で A3-E に引き渡すため，通常使用している (図 5 右側の) A から I のディスクを A3-E から切り離し，A6-NS の使用していた M9710 (図 5 左側) の A から I のディスクを使用する．

6.5 ディスク構成

図 5 のとおり 16 台のすべての M9710 型ディスクはデュアル・ポート接続である．そして，このディスクを 4 台ずつ四つのバンクに分け，合計 8 個の DLP から接続する．このうち 6 個を A6-NS, 2 個を A3-E 上に配置し，8 台分のディスクを 2 台のホストから共有 (同時期にはアクセスできない) する．基本的に 6.3 節で述べたように，2 台の M9710 を対の構成にして，重要なファイルの入ったディスクはそれぞれのディスク装置にまたがり互いにミラー・ディスクの関係にする (表 7)．

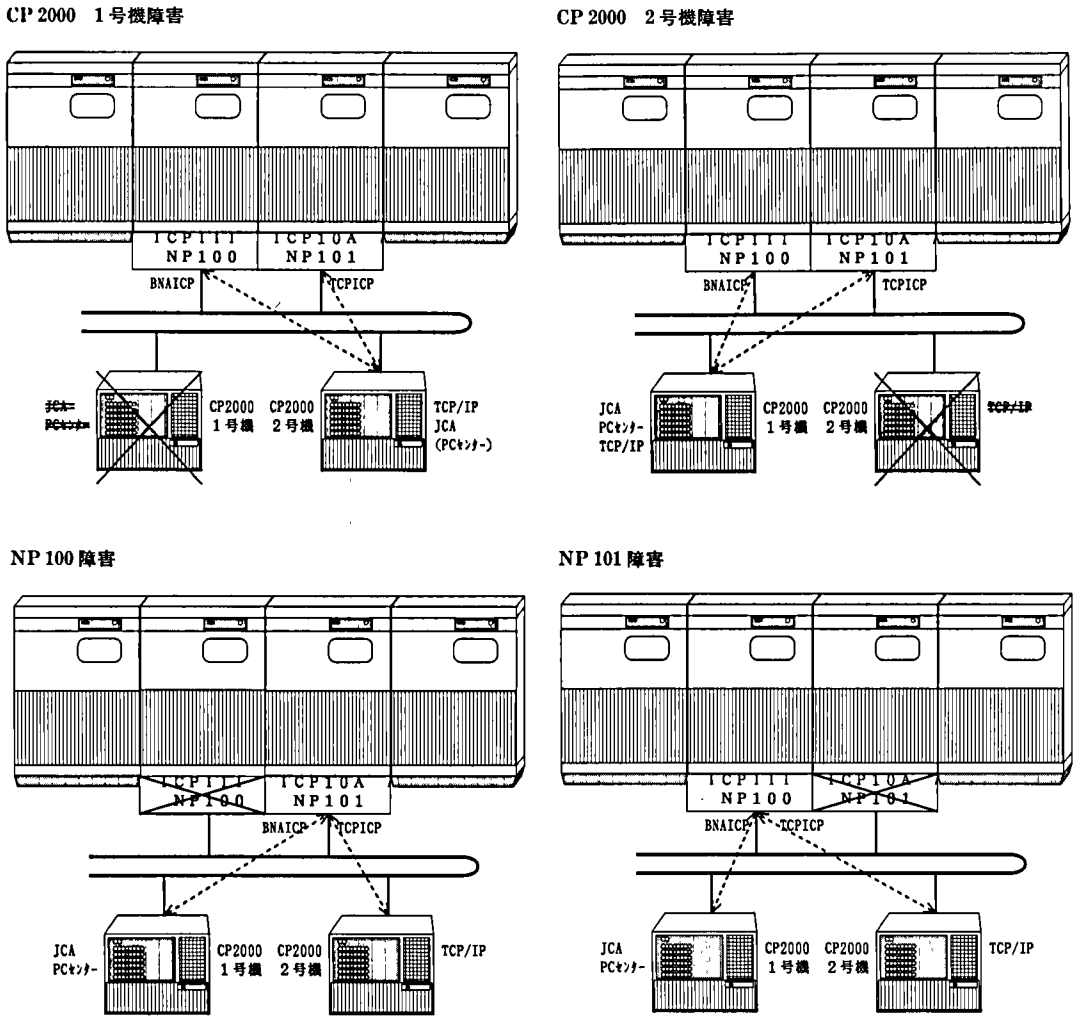
6.6 ネットワーク構成

O 社の構成は，先に述べたとおり EDC-DLP と ICP に接続した CP2000 の 2 種類の通信制御装置をもつ．標準端末装置と EOS 回線および光学文字読み取り装置 (OCR) を EDC-DLP から，UNIX マシンとの LAN 回線と VAN センタへの交換回線を CP2000 から接続している．PC センタへのスーパー・デジタル (SD) 回線も本部の CP2000 から接続し，PC センタ側にも CP2000 を設置している (図 4, 図 7)．

まず，EDC-DLP の回線については，既設ハードウェアのフィールド・アップグレードであったこともあり，現状では四つの EDC-DLP がある拡張キャビネットに集中している．このキャビネットにある I/O ベースの障害を想定すると，複数のキャビネットに分散した方がより安全であり，今後への課題の一つである．

次に ICP と本部の CP2000 については，ハードウェアを二重化し障害に対処することにした．通常は，それぞれ 2 台の ICP と CP2000 を UNIX マシンとの TCP/IP 用と PC センタサービス兼 VAN センタとの JCA 用に使い分ける．障害の発生時に，障害のあるハードウェアを切り離し，障害の無い ICP と CP2000 に全ての処理を集中して行わせる (図 6)．そのために CP2000 のボード構成をほぼ同一にし，それぞれの障害に対応するためのイニット・ファイル (ネットワーク定義ファイル) を作成した (図 7)．

A6-NS システムの障害時には A3-E システムに切り換えてバックアップすることになるが，構内回線は端子盤でのコネクタのつなぎ替え，その他の回線は回線スウィ



ICPIII, ICP10A : ICPの種類
JCA : 通信プロトコル(VAN, 銀行オンライン)
BNAICP : TCP/IPを使用しないCP2000のためのICP

NP100, NP101 : ICPの装置番号
TCP/IP : 通信プロトコル(UNIXマシン・ネットワーク)
TCPICP : TCP/IPを使用するCP2000のためのICP

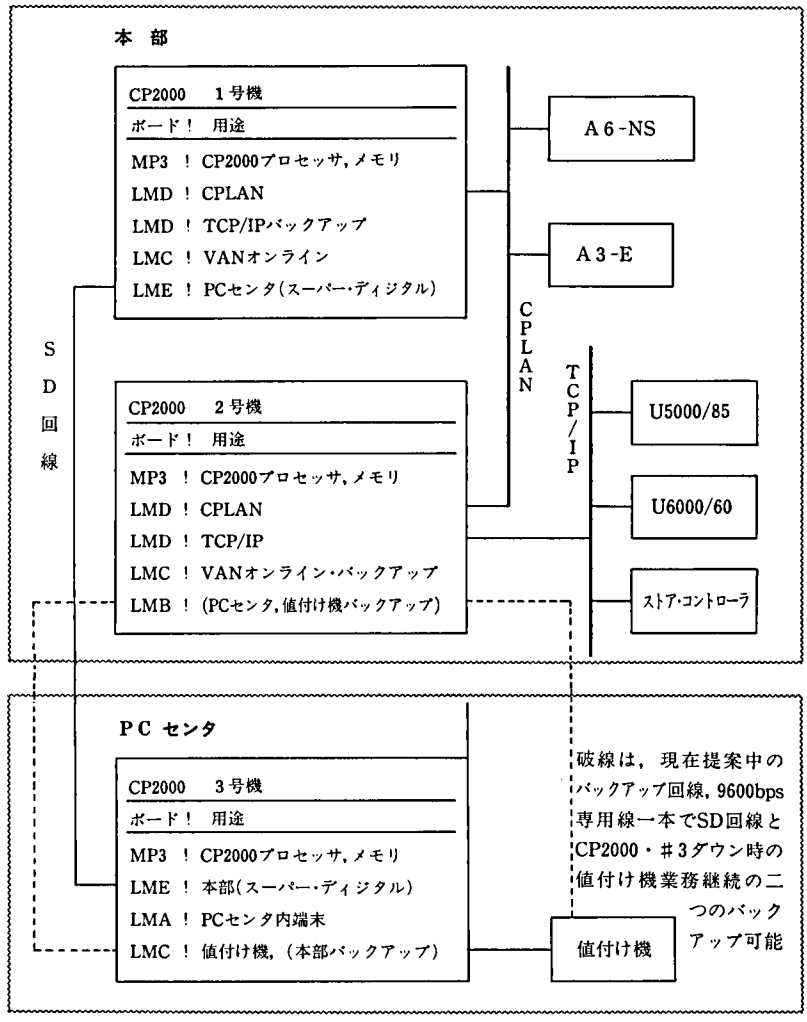
図 6　ICP, CP2000 障害対策

ッチング・ユニットの切り替えだけで回線の移動ができるようにした.

7. 障害時の対応

7.1 障害の特定

障害が発生すると，オンライン・サービスの利用者に応答がなくなり，システム担当者に連絡が入る．担当者は端末操作を通じて，それがアプリケーション・ソフトウェア上の問題かハードウェアの問題かを切り分けることになる．それが致命的な障害であればシステム・ダンプに陥るかシステム・ストップする．このとき基本的にはユニシスの技術員がダンプの解析やシステム自身の持つ自己診断機能の表示する診断コ



(注)ボードのタイプは，電気的インタフェースおよびボードの数により異なるが詳細は割愛する．

図 7 CP2000 ネットワーク

ードにより障害箇所を特定する．

それ以外の徴候からシステムの異常を検知したとき，システム・ログを解析したりODT 命令での装置状態の検査により障害箇所を特定する．

7.2 構成変更の操作

障害箇所が確認できたとして，それが CPU, IOA など根幹に関わる場合は，システム構成を変更する操作が必要になる．

1) PC-ODT 上のグループ (CPU, IOA などの組み合わせを定義した構成名) の変更

故障した側の CPU 系の切り離しをするために，適切なグループを選択する．

2) パワー OFF

システム停止手順に従いマシンのパワー OFF を行い，ブレーカを切る．

3) パワーON

障害のあるキャビネットを除いて電源を立ち上げる.

4) 構成の変更

ソフトウェア構成ファイル上の適切なグループを選択してRECONFIGURE*する（ホルトロードが伴う）.

この一連の操作だけで，30分程（ディスク・ファイルが大量にあるため，内約10分はディレクトリの作成）を要する．このため障害回復が構成変更を伴うような場合のMDTの目標を1時間以内に設定し顧客の了承を得た．特定の人だけしか回復操作ができないのでは障害発生から回復までの時間を遅らせる原因になりかねないので，それぞれのケースを想定して，考え得る障害箇所に対し，ダウン対策手順書を整備した．

表8　構成要素障害分析(ハードウェア)

キャビネット		基本キャビネット#1						基本キャビネット#2							拡張キャビネット#1							拡張キャビネット#2								A6
障害箇所(DLP) / 対象ユニット		キャビネット全	CPU	IOA	SCSI1	ODT2	ICP1	キャビネット全	CPU	IOA	SCSI1	ODT2			キャビネット全	SCSI1	LBP	TP3	SCSI1			キャビネット全	SCSI1	NSP5	NSP5	MTFIP	SCSI1	NSP5	NSP5	ICP1	A6全体
ディスク	DISK	P	P	P	P																	P	P			A				P
	(A)	P	P	P	P																	P	P			A				P
	(B)	P	P	P	P																	P	P			A				P
	(C)	P	P	P	P																	P	P			A				P
	(D)							P	P	P	P				P	P														P
	(E)							P	P	P	P				P	P														P
	(F)							P	P	P	P				P	P			A											P
	(G)							P	P	P	P				P	P			A											P
	(H)														X				X											P
	(I)														X				X											P
LBP															X		X													X
LP															A			A													A
MT																						B			B					A
NP100		A					A																							A
NP101																						A							A	A
DC112																						X		X						B
DC113																						X			X					B
DC114																						X					X			B
DC115																						X						X		B

X：該当ユニットが使用できずかつ代替要素がない.　　(注)DLP名称は表6注釈参照

A：代替構成要素を使用し業務は続行可能

P：パフォーマンスに影響が出るが該当ユニット使用可能

B：適用業務の一部機能を停止し代替構成要素で重要度の高い適用業務を続行

*RECONFIGURE：AシリーズのODTコマンドでグループの変更操作を行う.

8. 障害対策の評価

8.1 障害とその影響

構成要素の障害が，各ハードウェアに与える影響を 'X'（代替機器なし），'A'（代替機器で業務続行可能），'P'（パフォーマンス低下はあるが使用可能），'B'（重要な業務のみ実行）の各状態別に一覧にしたのが表 8 である．障害が CPU, IOA, BCC, I/O ベースのいずれかであれば，その影響はキャビネット全体となるため，表中には "キャビネット全"の欄で表現している．A6-NS 自体まったく使用できないケースを表中右隅の "A6" で表している．この表の中で障害に対し 'X' が示されているのが，対象とするハードウェアが使用できず，かつ代替要素がないケースである．ほとんどのケースにおいて 'A' または 'P' が示され，ほぼバックアップができる構成が確立されている．

8.2 障害の適用業務に与える影響

同様に，構成要素である各ハードウェアの障害が，業務に与える影響を一覧にしたのが表 9 および表 10 である．拡張キャビネット#2 にネットワークの制御が集中しているためキャビネット障害時に代替要素がないもの(NSP5＝EDC-DLP)がある．この業務（特に EOS）の優先度によっては A6-NS の障害と判断して A3-E を代替要素としてホストの切り替えを行うことになる．

表 9　構成要素障害分析(適用業務/DLP)

キャビネット		基本キャビネット#1							基本キャビネット#2							拡張キャビネット#1							拡張キャビネット#2								A 6
障害箇所(DLP) / 対象業務		キャビネット全	CPU	IOA	SCSI1	ODT2	ICP1		キャビネット全	CPU	IOA	SCSI1	ODT2			キャビネット全	SCSI1	LBP	TP3	SCSI1			キャビネット全	SCSI1	NSP5	NSP5	MTFIP	SCSI1	NSP5	NSP5	ICP1	A6 全体
オンライン	EOS	P	P	P	P				P	P	P	P				P	P			P			X	P	A	A		P				B
	PC センタ	P	P	P	P		A		P	P	P	P				P	P			P			P	P				P				B
	OCR	P	P	P	P				P	P	P	P				P	P			P			X	P	X			P				X
	VAN JCA	A	P	P	P		A		P	P	P	P				P	P			P			P	P				P				B
	銀行 JCA	A	P	P	P		A		P	P	P	P				P	P			P			P	P				P				B
	TCP/IP	P	P	P	P				P	P	P	P				P	P			P			A	P				P			A	B
	FAX 発注	P	P	P	P				P	P	P	P				P	P			P			X	P				P		X		X
	入力業務	P	P	P	P				P	P	P	P				P	P			P			X	P				P	A	A		B
テープ交換		P	P	P					P	P	P					P	P			P			X	P			X	P				B

X：適用業務が停止しかつ代替要素がない．
A：代替構成要素を使用し適用業務は続行可能
P：パフォーマンスに影響が出るが適用業務は続行可能
B：適用業務の一部機能を停止し代替構成要素で重要度の高い適用業務を続行

8.3 今後のシステム構成への課題

通信回線および LAN 回線は二重化していないため，障害時は回線の回復を待つことになる．とくに LAN 回線と SD 回線のトラブルは，PC センタのオンライン・サービスの停止につながるため，今後の補強の第一課題である（表 10）．

A6-NS 本体の障害や拡張キャビネット#2 の障害時の EOS のバックアップなど，

表10　構成要素障害分析(適用業務/CP 2000)

対象業務＼通信回線，CP 2000 障害	CPLAN	CP 2000 #1					TCPIPLAN	CP 2000 #2					SD回線	CP 2000 #3			
		MP3	LMD	LMD	LMC	LME		MP3	LMD	LMD	LMC	LMB		MP3	LME	LMA	LMC
VAN JCA	X	A	A		A												
銀行 JCA	X	A	A		A												
TCP/IP	X						X	A	A	A							
PC センタ	X	X	X			X							X	X	X		X
PC センタ改善案	X	P	P			P							P	P	P		A

(注) 表中X,A,Pは表9注釈参照
通信回線，CP 2000 は図7参照
PC センタ改善案は，本部と PC センタの間に 9600 bps 回線をバックアップ回線として敷設する案，これによりCPLAN障害以外のトラブルに PC センタ業務が受ける影響を免れることができる．

代替構成要素を開発用の A3-E に切り換えることで対処しているケースがいくつかある．弊社の表示しているプロセッサ能力の相対性能比較（RPM：Relative Performance Measurement)によると，A3-E は A6-NS の 9 から 10 分の 1 程度の能力であり，メモリも業務が動くにはかなり不足している．とくに EOS 発注など短いサイクルでアクセスのあるオンライン業務は，トラブル発生時に即座に代替手段をとりたいところだが，A3-E との能力差を考えると切り換えを躊躇せざるをえない．顧客側の設備投資予算との兼ね合いもあるが，ソフトウェアの構造や，A シリーズのリモート・データベース・バックアップ機能*の検討，EOS 環境，LAN 環境，待機系システムの能力アップなど今後に向けていくつかの課題がある．

また，現在のところ，障害箇所の特定および障害発生時に障害の修復とシステム構成の変更による障害の回避とどちらをとる方がより早く問題を解決できるかなど，技術員の判断を待つことになる．各ハードウェアのバックアップができるようなシステム構成が構築された後の課題は，障害対処において人的判断を少しでも減らすことである．現在この点を改善するためには，A シリーズの新しいソフトウェアである IOF** の障害監視サブシステムの適用が最適であると考えている．障害監視サブシステムは，以下のことを目的に作成されている．

1) システム内に発生する事象を定常的に監視し，システムが連続的に稼働する上で困難かつ危険な状態に陥っていないかどうかを判断する．
2) 危険な状態が発生したと判断した場合，利用者に対しこれを報告する．
3) 利用者による，稼働状況の悪化を防ぐための回避処理の実行を支援する．

9. おわりに

A シリーズの中型汎用機 A6-NS システムの分割運用機能を利用した障害対策事例を述べた．フォールトトレラント・コンピュータ（ノンストップ・コンピュータ）は，

*リモート・データベース・バックアップ機能：データベースの更新情報を，リアルタイムに遠隔ホストに用意したバックアップ用データベースに送信し，データベースのリモート・バックアップを可能とした機能．
**IOF：ユニシスの提供する統合運用システム，現在ジョブ管理・プリント管理・リソース管理・障害管理の各サブシステムが提供されている．

これまでとくにシステムの信頼性を要求されるシステム分野において使われてきたコンピュータであった．しかし，システムの高信頼性を望まないコンピュータ・ユーザは存在しないであろう．障害を局所化して業務を継続する技術がA6-KS/NSシステムで実現されたことにより，中小規模システムにおいても耐障害性の高いシステムが構築できるようになった．現在Aシリーズの製品ラインアップは，A7, A11, A16E, A19の4機種に統合されている．マイクロAからA6の後継としてA7, A6上位機以上の後継としてはA11が考えられ，分割運用機能はA11-222以上のマルチ・プロセッサ機からとなっている．基本的には本稿の考え方を，そのままA11システムに割り当てて考えることが可能である．また，A1, A4, A6システムからは，A6-KS/NSへのアップグレードの道も残されている．

今後，市場のニーズにともない24時間365日運転のシステムはますます増えることであろう．本稿が他の中小規模システムの信頼性向上の検討の一助になれば幸いである．

末筆ながら，貴重なご意見を賜ったユーザの方々，並びに弊社関係者各位にお礼を申し上げたい．

参考文献 ［1］ 日経コンピュータ 1993 6-21 増刊号「ハードウェア総覧 '93」日経BP社，1993.
［2］ (財)日本情報処理開発協会，「情報化白書 1992」，コンピュータ・エージ社，1992, pp. 240～241, pp. 379.

執筆者紹介 小 池 卓 (Takashi Koike)
1961年生．1984年 明治大学経営学部経営学科卒業．同年日本ユニシス(株)入社．LINCによる業務パッケージの開発を経て，フィールドにて流通業界のSEサービス，主にネットワークをはじめとする技術支援に従事．1994年1月より，システム技術第二本部 利用技術二部データコミュニケーション技術課に所属．

オープン環境化における異機種・分散システムの運用管理体系「ASMF」

1. はじめに

日本ユニシスは，90年代後半から21世紀に至る時代変革の中で，要求や変化に迅速に対応できる情報技術基盤を実現するための指針としてUA(Unisys Architecture)を提唱している．

ここでは，「分散化された異機種コンピュータ環境において広範囲な多種多様な利用者が，容易に必要な情報を取り出し，加工，活用できる環境」が今後求められる情報技術基盤であるととらえ，これをインフォメーション・ネットワークと名付けている．各企業における情報技術の広範囲な積極的な活用と一方では絶えることのない技術革新のめざましい進展は，これを現実のものとし，具体的な姿を現し始めている．

日本ユニシスは，このような環境を安全かつ効率よく運用，管理していくための体系（フレームワーク）であるASMF(Advanced Systems Management Framework)を発表した．

2. 背景－新しい運用管理のパラダイム

コンピュータの利用分野の拡大はオープン化，ダウンサイジング化といった流れを生み，運用管理の考え方を大きく変えざるを得ない状況になってきた．中央集中型のコンピュータシステム管理とネットワークの管理だけで運用管理を行っていた時代は，運用管理のエキスパートがすべてを網羅し，管理することが可能であった．しかしながら，今日のようにWANやLANが網の目のように張り巡らされ，コンピュータシステムが地域的に分散し，その構成もマルチベンダによる異機種コンピュータ，高機能なインタネットワーキング機器(PBX，ブリッジ，ルータ等)，さまざまな回線形態，通信プロトコル等の混在する中での障害の検出や切り分け，復旧指示は困難になっている．

日本における分散システム化は，情報活用系システムの分野を中心として，小規模から大規模まで数多くの事例が報告されている．一方，企業の基幹中枢業務系の分散化は，経営戦略からの要請として，今まさに離陸の段階といえる．その実施にあたり最も懸念されるのが，分散されたデータ，プログラムおよび機器類を，安全かつ適切に運用できる技術の確保である．従来のホストコンピュータを中心とした運用管理では，せいぜい10台に満たない数の管理対象であったものが，UNIX*やPCがLANに接続された瞬間に数百から数千台を対象とした運用管理となる．ここでは従来の運用管理の考え方や仕組みでは，対応できない．そして，この新しいパラダイムを実現するためには，それを支える技術，仕組みが必須となる．

3. 新時代の運用管理に対する要請

これからの運用管理に対して求められるものは，今までの運用管理の要件に加え，新たな観点からの要件を必要とする．

- 変化する要求に柔軟な対応ができる運用管理

  今日のような激動の社会情勢では，ビジネスの変化は急速であり，それを支える企業の情報技術基盤も定常的なものでは有り得ない．変化するハードウェア，ネットワークおよびソフトウェアを把握・制御できる新しい運用管理の考え方が必要である．

- 分散処理環境での運用管理手段

  オープン化，ダウンサイジング化の大きな目的はダウンコスティング，すなわちコストの削減である．しかし統合的な指針や枠組みのないケースにおいては，逆に多大なコスト負担を引き起こす可能性が高い．分散処理環境を前提とした統合的な運用管理が強く求められる．

- 既存システムとの連携

  コンピュータ・システムがすでに稼働している場合には，運用管理システムが大なり小なり存在している．ここには各企業におけるノウハウや業務や組織と密接に結びついた運用管理が行われている．この既存資産（ノウハウ）を生かすために，これらを継承しつつ，新しい運用管理と連携できることが重要とな

* UNIXオペレーティングシステムはUNIX System Laboratories, Incが開発し，ライセンスしている．

る．

・国際標準/業界標準への準拠

多機種にまたがる運用管理を統合的に行うには標準規約にそった技術や仕組みの採用が必須である．これによって始めて，マルチベンダ環境への対応が現実に可能となる．

4. ASMF

ASMFは，上記の背景を踏まえ，時代の要請に応える『マルチベンダ環境におけるインフォメーション・ネットワークを統合的に運用管理するためのフレームワーク（製品体系）』である（図1）．

4.1 ASMFの基本的な考え方

ASMFでは，以下のような考え方をもとに運用管理に対する新しい体系をまとめた．

1) マルチベンダ対応

ISOやX/Open*，OMG(Object Management Group)等の国際標準，業界標準の技術を積極的に採用する．これにより異機種を含んだマルチベンダ環境における統合的な運用管理を実現していく．

2) 統合化

各企業のそれぞれの実態にあった運用管理を構築するため，既存の管理システムや，ユニシス以外の他ベンダの管理システムとの協調・連携を行う統合的な仕組みを提供する．

3) 自動化

運用操作上でのコストの低減や人的なミスを防ぐために，オペレーション作業の自動化を推進する．

4) 最新技術の採用

継続的に発生する変更に対し，柔軟な対応や分散システム化された環境への迅速な適応のために，最新技術であるオブジェクト指向技術やクライアント/サーバ技術の積極的な採用を行っていく．

4.2 ASMFで採用する主な技術

ASMFでは将来の技術革新にも対応できるよ

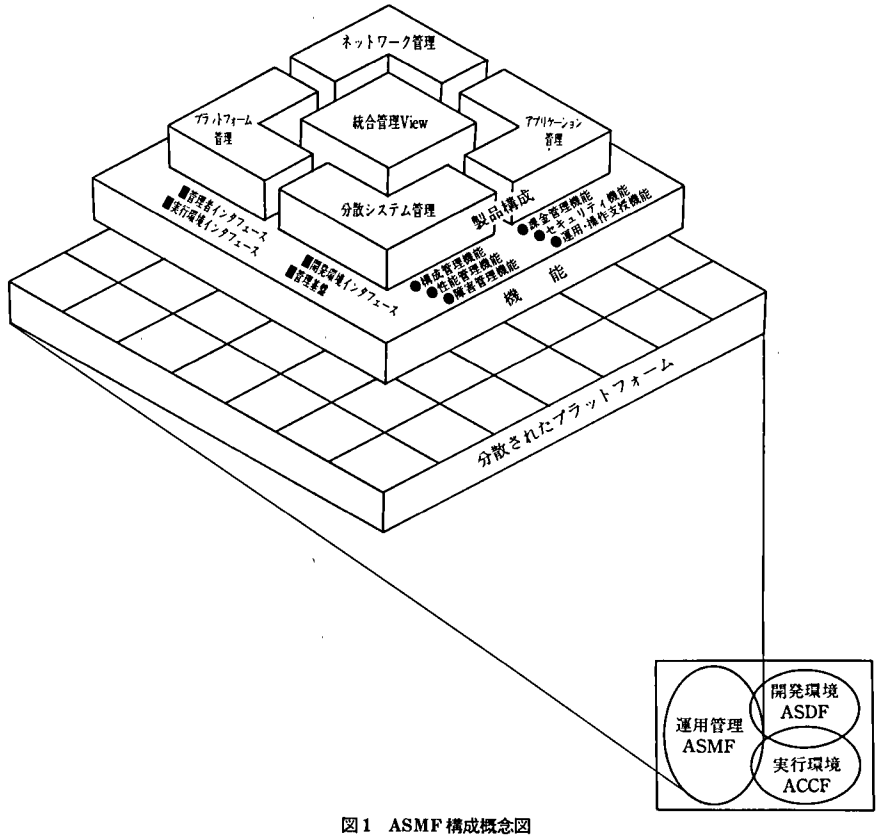


図1 ASMF構成概念図

* X/Openは X/Open社の登録商標である．

うに，次のような先進技術を積極的に採用する．

1)　オブジェクト指向技術

運用管理対象の多様化，システム構成の変更への柔軟な対応のため，柔軟性・拡張性の実現に最も適した技術としてオブジェクト指向技術を採用する．とくに様々な管理機能の相互連携を実現するための基盤部分にオブジェクト指向技術の標準化推進団体であるOMGが提唱している「CORBA(Common Object Request Broker Architecture)」に準拠した「ORB」(Object Request Broker)を採用していく．

2)　GUI(Graphical User Interface)

人的ミスを防ぎ，容易で効率的な運用管理を実現するため，管理対象の機器構成などをグラフィック画面を用いたわかりやすいインタフェースで表示する機能を提供する．ここではMotif*などのGUI技術を使用する．

3)　標準管理プロトコルの採用

統合的な運用管理を実現するため，同機種間や異機種間の情報交換を行う管理プロトコルとして国際標準/業界標準の「CMIP」(Common Management Information Protocol)および「SNMP」(Simple Network Management Protocol)などを採用する．

4)　セキュリティ

分散化された情報処理環境の高い機密性や安全性を保証するため，標準の規約に基づいた認証サービスとアクセス制御を採用する．

5)　エキスパートシステム

地域的に分散された管理対象の自動運転や障害時の故障診断での原因の発見や的確な対応を実現するため，先進のエキスパートシステムを活用する．

6)　マルチメディア

的確な情報の伝達や迅速な判断を支援するために音声のみならず，マルチスクリーン，動画などのマルチメディア技術を積極的に取り入れていく．

5. ASMFの機能

ASMFは運用管理に関する機能を構成管理，障害管理，性能管理，課金管理，セキュリティ，運用・操作支援の六つに分類し，これらの機能と運用管理者，開発環境，実行環境との連携をとる三つのインタフェースを設定している．さらに，これらを相互連携するための「管理基盤」により構成されている(図2)．

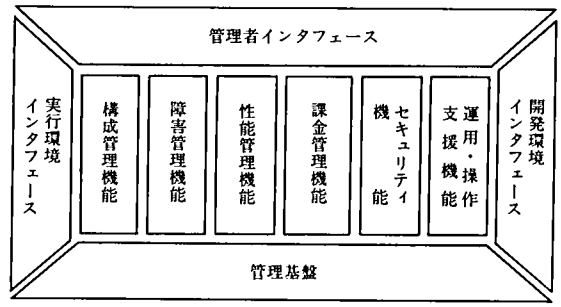


図2　ASMFの機能

5.1　管理機能概要

1)　構成管理機能

ハードウェア/ソフトウェアの構成やそれらの管理情報を持ち，ソフトウェアの配布・適用などを行う．

- 資材管理：システム資源の配置，運用構成の追跡・制御
- 導入計画：導入・構成変更のスケジューリング，構成の生成
- 導入：事前に配布された構成変更のシステムへの導入
- 配布：ハードウェア/ソフトウェアの自動・手動配布
- 構成：分散したハードウェア/ソフトウェアを環境に適合させるためのカスタマイズ
- ライセンス管理：ソフトウェアの契約形態に基づく管理

2)　障害管理機能

障害発生の監視，報告支援，予防保守（点検などの管理サイクルの管理）を行う．

- 予防：障害管理情報に基づいた統計分析などによる障害の予防
- 検出・ロギング：管理対象の監視，異常事象の検出および事象の記録
- 追跡・報告：障害の追跡と障害に関する情報の報告
- 診断・特定：パターン認識，障害の相関，診断テスト機能を使用した障害の特定および診断
- 回復：障害を起こした資源が提供していたサービスの復元

* Motifは Open Software Foundationの登録商標である．

3) 性能管理機能

管理対象の定義，監視，分析/評価，報告支援，管理対象物の容量/余裕などのキャパシティ・プランニングを行う．

- ●管理対象定義：性能データを収集する管理対象資源の定義
- ●監視：性能を測定するためのデータの収集，システムの利用状況の把握
- ●報告：性能データの報告
- ●分析・調整：性能測定値の評価と性能改善への処置
- ●キャパシティ・プランニング:ユーザ要求について費用対効果の高い計画策定支援

4) 課金管理機能

使用実績収集，それに伴う計算，報告（請求）を管理する．

- ●計測：資源使用量に対する課金情報の記録・蓄積
- ●報告：課金記録の伝達と報告
- ●制御：計測パラメタ，課金記録の制御

5) セキュリティ機能

認証/アクセスに関する登録，監視制御などの機能によりシステムの機密を保護する．

- ●監視：セキュリティに関係する処理の識別と記録
- ●制御：管理対象資源に対する操作時の制御
- ●報告：セキュリティに関連する処理の報告

6) 運用・操作支援機能

インフォメーション・ネットワークにおける資源の信頼性と可用性の向上を目的とした，ベンダ，装置タイプに依存しない一様な資源運用の支援機能を提供する．

- ●自動運用：スケジュール，イベントなどに基づいた自動的なアクションの実行
- ●ジョブ管理：ジョブの管理，自動運行，監視
- ●バックアップ/リストア：計画的なファイルのバックアップと管理操作によるファイルのリストア（回復）
- ●装置制御：各種装置に対する管理操作
- ●稼働統計：稼働データの記録と報告

5.2 インタフェース

運用管理環境は以下の三つのインタフェースを設定している．

1) 管理者インタフェース

管理情報の表示/管理者コマンドの入力処理など

2) 実行環境インタフェース

実行環境との連携を図るためのインタフェース

3) 開発環境インタフェース

開発環境との連携を図るためのインタフェース/ユーザ固有管理アプリケーションの構築支援ツール群

5.3 管理基盤

管理基盤は管理対象の定義と，それらに対する操作と通知（応答）などの情報によって構成される情報管理ベースと管理情報の伝達をするための仕組みをオブジェクト指向技術を使って提供する(図3)．

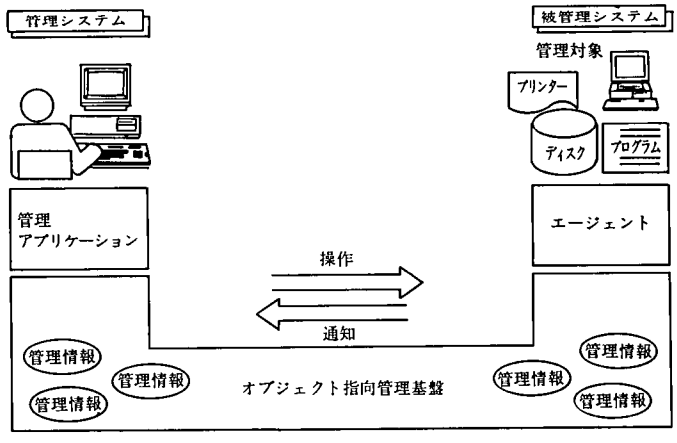


図3 管理基盤概念図

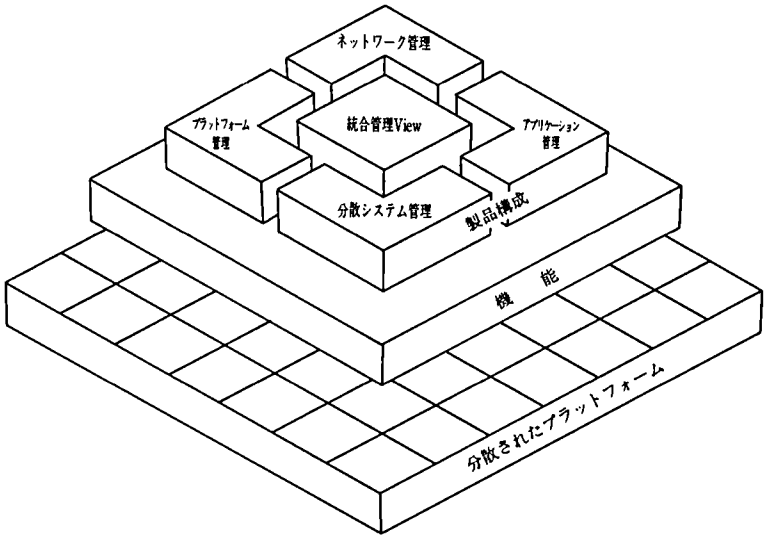


図4　ASMFの製品構成

6. 製品構成

ASMFでは各種の運用管理製品を管理対象の内容によって「分散システム管理」「アプリケーション管理」「プラットフォーム管理」「ネットワーク管理」の四つの管理分野に分け，さらにこれらの統合的なインタフェース「統合管理View」を加え，情報システム管理に関する製品群を体系化した（図4）．

注：機能と製品構成の関係

前記の機能とこれら四つの管理分野との関係は独立であり，機能はそれぞれの分野にすべて存在するが，その重要度や構成比率は分野ごとに異なる．

6.1　プラットフォーム管理

個々のコンピュータ・プラットフォームに着目した管理分野で，一つまたは同種・複数のコンピュータ・システムで構成される閉じた管理システムである．この分野における汎用機を中心とした管理システムは歴史も古く，機能面で最も充実している．しかし，UNIXプラットフォームを対象にしたものは，いまだ十分とは言えない状態にある．今後早急に充実する必要のある分野である．

6.2　ネットワーク管理

ネットワークに着目した管理分野を指す．LAN/WANを中心としたネットワークに関する構成，障害，性能などを主たる機能とする．分散処理環境では，ネットワークの重要性が高まっており，ネットワーク管理は運用管理において必須のものとなりつつある．

6.3　アプリケーション管理

この概念はASMFではじめて提唱するもので，業務アプリケーションに着目した管理分野を指す．このアプリケーション管理では，業務アプリケーションと，それに関係する各種のデータ，使用機器，使用者，利用形態などの種々の業務アプリケーションとの関連を管理する．これによって，たとえばシステム内に，ある障害が発生した場合，その障害が及ぼす業務アプリケーション範囲を特定し，復旧の方策やユーザへの通知など，迅速な対応を支援する．

またシステム構成の変更に対する業務アプリケーションへの影響などに対しても正確・迅速な対応を可能にし，業務処理の円滑な運行を支援する．

6.4　分散システム管理

WAN/LANのネットワークに接続されている分散された各種（異機種を含む）のコンピュータおよびネットワーク機器により構成されているシステムを対象とした管理分野を指す．オープン化，ダウンサイジング化の急速な進展により，企業内におけるシステム環境は今までと比較にならないほど複雑化してきている．このため構成情報の管理やソフトウェアの配布など企業情報ネットワークの円滑な運用を支援する管理システムへのニーズが高まっている．

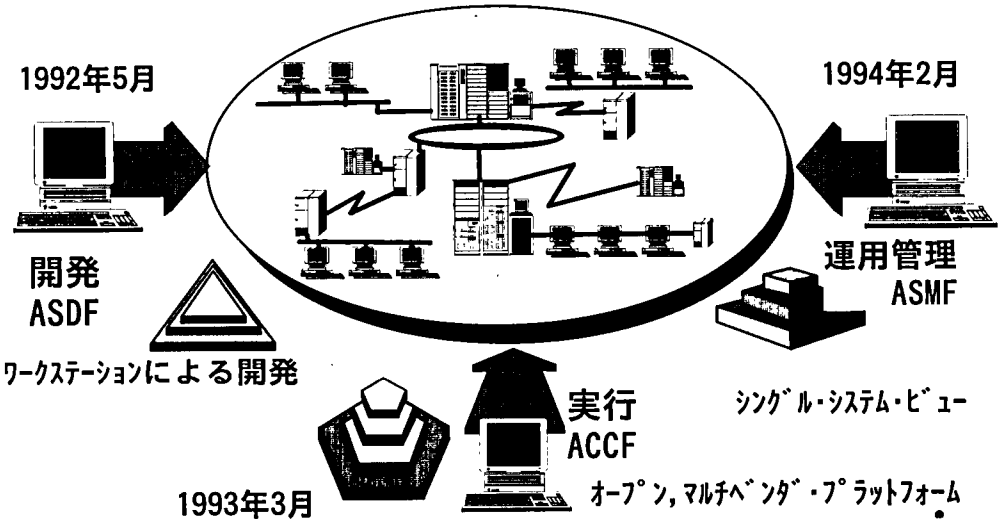


図5 UAの展開

この分散システム管理では，アプリケーション管理，プラットフォーム管理，ネットワーク管理との協調および多種多様な機器の管理のために国際標準/業界標準への積極的な準拠が必要である．

6.5 統合管理 View

各管理システムのヒューマン・インタフェースを統合的に行うモジュール．現在，各々の管理システムは独自のインタフェースを持ち，操作法，操作端末も異なっているのが現状である．

このため，管理対象の種類や数量が増大するに伴って，それぞれの管理用ハードウェアを揃えたり，各々の操作方法を習得しなければならず，運用効率が著しく低下することになる．

ASMFでは，操作の容易性と正確性を実現できる統合的な管理インタフェースを「統合管理View」と呼び，シングル・システム・ビューを目指す．

7. おわりに

日本ユニシスではUAに基づき，開発環境での体系ASDF(Advanced Solution Development Framework)を92年5月，分散協調実行環境での体系ACCF(Advanced Cooperative Computing Framework)を93年3月に発表してきた．そして今回運用管理環境の体系であるASMFの発表により，インフォメーション・ネットワークを具体的に実現するシナリオが完成した．

日本ユニシスでは今後これらの体系に基づいた商品を継続的に提供することにより，開発者，利用者，運用管理者など情報処理に関与する人々が，システム構築，活用，管理など各々の仕事において高品質，高生産性，高効率に遂行できる環境を実現する（図5）．

掲載論文梗概

オペレータはコンソールメッセージをもとにバッチラン進捗状況を管理しているが，ラン終了時刻等の予測は勘で行っている．このため，バッチラン終了予測，スケジュール作成を行う運用支援システムが求められている．松田芳雄・河岸憲一の**バッチラン運用管理システムにおける所要時間の予測とスケジュール作成**は，2200/1100 シリーズにおけるバッチラン所要時間の予測方法を紹介し，最適スケジュール作成方法を述べている．

ニューラルネットワークの学習アルゴリズムとして提案された BP 学習アルゴリズムについて，学習速度の高速化・局所最小値への落込みの解決法がいくつか提案されている．浦上浩一の**ニューラルネットワークにおける学習の高速化と収束安定性の考察**では，GA 学習法，最適初期値設定法，局所最小値回避法，高速 BP 学習法を取り上げ，実問題での実験・相乗効果の検討を行っている．

国際テレックス電文の解読処理を人工知能の手法を使って自動化した．中田純一の**知識ベースを利用した国際テレックス電文の自動解読**では，限定的ではあるがテレックス文という自然語文をコンピュータにより解読することを実現した三和銀行テレックス電文解読システムの技術的成果について，開発を担当した立場から報告している．

日刊スポーツ新聞社の案内広告システムは同社と広告代理店とを接続して EDI による案内広告の集配信を実現し，新聞制作上のネックとなっていた配列・割付工程を AI を活用して完全自動化を実現している．近藤千秋は**日刊スポーツ新聞社における「案内広告システム」**の中で，その実現に至る経緯と技術的先進性を紹介している．

流通業界では競合他社との差別化を図るための顧客サービス競争の結果，システム規模の拡大や長時間連続運転への期待が大きくなっている．小池卓は **A6-NS による高信頼性システムの構築**の中で，総合スーパーO 社の A 6-NS システムの導入事例を通して，その分割運用技術の利用と耐障害性を高めたシステム構成を紹介している．

掲載論文梗概

▶技報編集委員会
委員長　柳生孝昭
副委員長　小林　允
委　員　朝倉文敏，古村哲也，丸山　修
内藤　聰，岩佐宏一，深堀年弘
松倉　司，西原惠二，樫山　汎
大桃　忠，河本太都夫，育柳幸久
木村修三，久保田俊雄，村岡俊彦
馬場正存，鎌田　稔，大高哲彦
高畑和夫

▶編集制作担当
システム企画部 標準企画室
駒崎洋介，丹野敬子
総合マーケティング部
熊谷　貴

● Editorial Board
T. Yagiu (Chairman)
M. Kobayashi (Vice Chairman)
F. Asakura, T. Komura, O. Maruyama
S. Naito, K. Iwasa, T. Fukabori
T. Matsukura, K. Nishihara, H. Kashiyama
T. Omomo, T. Komoto, Y. Aoyagi
S. Kimura, T. Kubota, T. Muraoka
M. Baba, M. Kamata, A. Otaka
K. Takahata

● Editorial Staff
Y. Komazaki, K. Tanno
(Systems Operations Planning)
T. Kumagai
(Corporate Planning & Marketing)

ISSN 0914-9996

技　報
UNISYS TECHNOLOGY REVIEW
Vol. 14　No. 1　(No. 41)

発　行　日　平成6年5月31日
編集発行人　柳　生　孝　昭
発　行　所　日本ユニシス株式会社
東京都江東区豊洲 1-1-1 〒135
TEL(03)5546-4111 (大代表)
印　刷　所　三美印刷株式会社

禁無断複製転載

© Nihon Unisys, Ltd. 1994

UNISYS

どうやってつながったんだろう。

❶四方かま継ぎ
柱に用いられる継ぎ手。
真横からはどっちの面からも
入りそうにない…。

❷箱栓
表を見るとただくっつけただけ。
ところが、実は裏側に細かい細工が。

❸十字めちいれ
十字に切り込みを入れて
組み合わせ、材のずれや
回転を防ぐための継ぎ方。

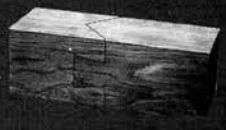
❹宮島継ぎ
斜めの切り口なので接合面が広く、
密着させるには高度な技術が必要。

❺貝の口継ぎ
塔の中心柱などに
使われる、接合面を
大きくし、強度を確保。

❻丸太桁
まるで2本の丸太ががっちり
組み合わされているようですが、さて。

❼腰かけはま継ぎ
土台などに使われ、
上からの大きな重量に耐えるように
考えられている。

❽二枚ほその仕口
上のクサビ型の部分だけでも難しいのに、
さらに下にもうひとつのホゾが。



❾大坂城大手門控え柱の継ぎ手
以前クイズにもなった、不思議な継ぎ手。

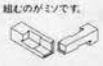
「つながるはずがない」を、つなぎます。ユニシスのオープン・システム・テクノロジ。

上にご紹介したのは、古くから木造建築で使われてきた「継ぎ手」と呼ばれる接合技術。まるでパズルのような不思議なものもあります。昔も今も、もの作りの基本は、組み合わせの技術。コンピュータ・システムも、異なるメーカーのさまざまなコンピュータをどう組み合わせ、最適にするかを考える、オープン・システムの時代。ユニシスはオープン・システムに早くから本格的に取り組み、そのノウハウの蓄積を活かして高品質のシステムを提供しています。そのひとつが、U6000シリーズをサーバにしたクライアント/サーバ・システム。異なるメーカーのワークステーションやパソコンを結んでデータの共有化を実現するのはもちろんのこと。ユニシスの場合は、そのクライアント/サーバ・システムを、いまお使いのホスト・コンピュータに、メーカーを問わず接続。ホストが持つ既存のソフトやデータをフルに活かすとともに、将来的にはホスト・システムのダウンサイジングをも可能にします。ユニシスが実現する、一歩進んだオープン・システム。その先進性を支えるU6000シリーズが、いまいちだんと性能を向上して新登場。全方位へつながるオープン・システム・テクノロジが、さらに広がりました。

➡こうやってつながりました。

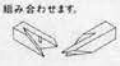
❶斜めの方向から
組み合わせるのが秘密。

❷両側からはさむように
組むのがミソです。



❸凸型と凹型の
正確さが決め手です。

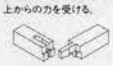
❹互い違いの切り込みを
組み合わせます。



❺1/4ずつの切り込みが
互い違いに。

❻強度を上げるために
ホゾ穴を最小限に。

❼腰を掛けたような状態で
上からの力を受ける。



❽ホゾ穴の斜めの
勾配にそって入れる。

❾斜めにずらしながら入れる。

ユニシス UNIX サーバ
U6000シリーズ
Pentium™プロセッサ搭載のU6000/300新登場

※UNIXはX/OPENリミテッドカンパニーが独占的にライセンスする、米国およびその他の国における登録商標です。
※Pentiumは米国インテル社の商標です。

資料提供：住吉 廣七・松井 源吾 共著 鹿島出版刊「木造の継手と仕口」

日本ユニシス株式会社　本社　東京都江東区豊洲 1-1-1 〒135 電話03-5546-4111(大代表)