０４－開－２１

データベース構築促進及び技術開発に関する報告書

グループウェアにおけるデータベースシステムに関する調査研究

平 成 5 年 3 月

財団法人　データベース振興センター
委 託 先　株式会社イフ・アドバタイジング

本事業は、日本自転車振興会から競輪収益の一部である機械工業振興資金の補助を受けて作成したものである。

序

データベースは、わが国の情報化の進展上、重要な役割を果たすものと期待されている。今後、データベースの普及により、わが国において健全な高度情報化社会の形成が期待される。さらに海外に対して提供可能なデータベースの整備は、国際的な情報化への貢献および自由な情報流通の確保の観点からも必要である。しかしながら、現在わが国で流通しているデータベースの中でわが国独自のものは1／3にすぎないのが現状であり、わが国データベースサービスひいてはバランスある情報産業の健全な発展を図るためには、わが国独自のデータベースの構築およびデータベース関連技術の研究開発を強力に促進し、データベースの拡充を図る必要がある。

このような要請に応えるため、㈶データベース振興センターでは日本自転車振興会から機械工業振興資金の交付を受けて、データベースの構築および技術開発について民間企業、団体等に対して委託事業を実施している。委託事業の内容は、社会的、経済的、国際的に重要で、また地域および産業の発展の促進に寄与すると考えられているデータベースの構築とデータベース作成の効率化、流通の促進、利用の円滑化・容易化などに関係したソフトウェア技術・ハードウェア技術である。

本事業の推進に当って、当財団に学識経験者の方々で構成されるデータベース構築・技術開発促進委員会（委員長　山梨学院大学教授　蓼沼良一氏）を設置している。

この「グループウェアーにおけるデータベースシステムに関する調査研究」は平成4年度のデータベースの構築促進および技術開発促進事業として、当財団が株式会社イフ・アドバタイジングに対して委託実施した課題の一つである。この成果が、データベースに興味をお持ちの方々や諸分野の皆様方のお役に立てば幸いである。

なお、平成4年度データベースの構築促進および技術開発促進事業で実施した課題は次表のとおりである。

平成5年3月

財団法人　データベース振興センター

平成4年度　データベース構築・技術開発促進委託課題一覧

分　野	課　題　名	委　託　先
社　会	1　変異タンパク質配列データベースの構築	日本電子計算㈱
	2　新聞縮刷版見出しデータベースの構築	㈱朝日新聞社
	3　ファジィに関する文献データベースの構築	(財) 日本情報処理開発協会
	4　医療用医薬品抗生物質データベースの構築	㈱小田島
	5　交通事故調査データベースの構築	(財) 日本自動車研究所
	6　楽器データベースの構築	㈱ダイソメディアサービス
	7　人体計測データベースの構築	(社) 人間生活工学研究センター
	8　大学におけるデータベース利用教育システムのプロトタイプ作成	日外アソシエーツ㈱
	9　先進複合材料データベースの構築	(財) 次世代金属・複合材料研究開発協会
	10　博物館所蔵地図資料所在情報データベースの構築調査	(財) 地図情報センター
中小企業振興地域活性化	11　地域流通最適化データベースのプロトタイプ作成	(社) 日本ボランタリー・チェーン協会
	12　異分野研究のための知的オリエンテーション・データベースシステムのプロトタイプ作成	㈱けいはんな
	13　在宅勤務者サポート・データベースの構築調査	㈱志木サテライトオフィス・ビジネスセンター
海　外	14　銅基複合材料日本特許英文データベースの構築	神鋼リサーチ㈱
	15　技術協力供与機材データベースのプロトタイプ作成	(財) 日本国際協力システム
	16　先端産業分野における専門用語の電子辞書データベース化の調査研究	科学技術情報研究所㈱
	17　マーケティングコードの英文データベースの構築	㈱帝国データバンク
技　術	18　安全研究における多重シソーラス・システム構築のための基本安全用語データベースの開発	㈱紀伊國屋書店
	19　3次元マッピングデータベースの技術開発	㈱日本総合技術研究所
	20　データベース検索サポートシステムの調査研究	セントラル開発㈱情報図書館ＲＵＫＩＴ
	21　グループウェアにおけるデータベースシステムに関する調査研究	㈱イフ・アドバタイジング
	22　パーソナルコンピュータとＬＡＮの利用による非定形データベースのプロトタイプ作成	㈱メイテック
	23　知的資源型データベースの調査研究	㈱ジャパンコミュニケーションズインスティテュート

目　　次

Ⅰ．課題性の研究

Ⅱ．グループウェアにおけるデータベースシステムの構築

1章　グループウェアの技術的動向…… 3

はじめに…… 3

1.　グループウェアの分類…… 5

2.　即時直接対面型のグループウェア…… 6

3.　即時遠隔分散型グループウェア…… 10
　3.1　グループエディタ…… 10
　3.2　ウインドウ共有／画面共有システム…… 12
　3.3　ワークステーション／ビデオ会議システム…… 14

4.　非即時型遠隔分散型グループウェア…… 16
　4.1　ビジネス・コーディネーションシステム…… 16
　4.2　議論支援システム…… 18
　4.3　オフィスワークフローシステム…… 20
　4.4　メッセージ構造化システム…… 21
　4.5　協同文書作成支援システム…… 23

2章　グループウェアにおけるデータベースシステム構築技術…… 28

1.　データベースシステムの技術動向…… 28
　1.1　データベースシステム…… 28
　1.2　リレーショナルモデル…… 29

2.　グループウェアデータベースシステムの基本技術…… 32
　2.1　オブジェクト指向データベースシステム…… 32
　　2.1.1　基本概念…… 32
　　2.1.2　オブジェクト指向プログラミングシステムの例…… 37

2.1.3　リレーショナルモデルでのオブジェクト指向…… 40
2.1.4　オブジェクト指向データベースシステム利用の効用…… 43
2.2　グループウェアにおけるデータベースシステムの自律性と協調…… 54
2.2.1　グループウェアにおける分散型データベースシステム…… 54
2.2.2　多データベースシステムの異種性、自律性、分散性…… 57
2.2.3　自律型分散データベースシステムの統合化…… 60
2.2.4　分散型データベースシステムの参照モデル…… 61
2.2.5　システム構成とトランザクション管理…… 64
2.3　グループウェアにおける分散型データベースシステムの設計…… 67
2.3.1　設計機能とスキーマ変換…… 67
2.3.2　グループ通信方式…… 68
2.3.3　分散型データベースのシステムの例…… 70

3.　グループウェアデータベースシステム…… 75
3.1　オブジェクト指向データベースシステム…… 75
3.2　分散型データベースシステム…… 75
3.3　利用者インターフェイス…… 76

Ⅲ. グループウェア発展に伴う情報システムの高度化ならびにデータベース構築技術の進展とその課題

1.　グループウェアの発展と情報システムの高度化…… 77

2.　グループウェアの発展とデータベース構築における今後の課題…… 78
2.1　グループウェア設計上の問題点…… 78
2.2　グループウェア利用上の問題点…… 79
2.3　グループウェアの統合化…… 80
2.4　協調理論の発展…… 81

Ⅰ．課題性の研究

1. 目　　的

これまでの情報システムは、個人の作業の支援が中心であったが、今後は、グループ作業の支援、いわゆる”グループウェア”が情報システムの主要な目的となる。しかし、グループウェアの遂行には様々な技術的課題の解決が必要であり、中でも各個人、グループで利用されるデータベースの実現、管理についての検討が極めて重要となる。つまり、各個人のデータベースシステムとグループデータベースシステムから構成される分散型データベースシステムについての詳細な調査・研究が必要となる。このため、グループウェアにおけるデータベースシステムに関し、特にユーザにおける情報システムの高度化を視点として、その技術と展望、ユーザにおける活用、課題等様々な側面から調査研究を実施し、データベース構築技術の向上を図るとともに、わが国におけるデータベースの発展に寄与することを目的としている。

1. 実施内容

本調査研究では、以下の項目を主たる内容としている。

1）　グループウェアの技術的動向
グループウェアについての考え方、グループウェアの分類、各種グループウェアの現状と技術的動向についての考察。

2）　グループウェアにおけるデータベースシステム構築技術について
グループウェアにおけるデータベースシステムの基本技術、データベースシステムが持つ特徴、グループウェアにおけるデータベースシステムの設計等についての調査研究。

3）　グループウェア発展に伴う情報システムの高度化ならびにデータベース構築技術に進展とその課題について

今後の情報化の進展に伴うオフィスワークにおけるグループウェアの重要性とそれを支えるデータベースの必要性、グループウェア発展に伴う解決すべき課題等についての考察。

1. 実施体制

株式会社イフアドバタイジング内に当該事業遂行のための事務局を設置し、このもとで「調査・研究会」を編成して、調査研究を実施した。

調査研究会メンバー

滝沢　誠（東京電機大学理工学部・経営工学科助教授）

山鳥雄嗣（㈶日本情報処理開発協会・調査部長）

市川　隆（㈶日本情報処理開発協会AI・ファジー振興センター所長）

道田國雄（新世代システムセンター企画部長）

近藤伸一（新世代システムセンター調査部長）

Ⅱ. グループウェアにおけるデータベースシステムの構築

1章　グループウェアの技術的動向

はじめに

近年、コンピュータシステムのパーソナル化や、それらを結び付けるネットワークシステムの普及によりグループウェアと呼ばれる新しいコンピュータ利用形態あるいはツールが注目されている。

グループウェアに関して、最初にこの用語を用いたとされているジョンソンーレンツは「グループウェアとは、コンピュータシステムに社会的グループのプロセスを付加したもの」と述べている[Lentz]。また、「グループウェア」の著者である米未来研究所のジョハンセンは「共同作業をするワークグループ専用に設計されたコンピュータ支援型システムの総称」と述べている［Johansen］。

従来コンピュータシステムは、データを正確かつ高速に処理するという観点から設計されてきた。その結果、込み入った計算、大量のデータ処理は、コンピュータで行われるようになり、これらの作業に携わる人々の労力は大幅に軽減されてきた。一方、新しいコンピュータの利用形態であるグループウェアは、グループにおける共同作業、問題解決、合意形成といった社会的プロセスそのものを支援しようとするものである。

他の言い方をすると、従来のソフトウェアはユーザが単にコンピュータの物理的パワーを享受するマン・マシーンの観点で設計されてきたが、グループウェアは、コンピュータを介してグループ内の作業が促進されたり、増幅されるよう設計されたシステムといえる。例えば、グループ意思決定支援システムは、グループの知識を結集、増幅させることにより、より良い意思決定を導こうとするものである。

グループウェアは、その内部に社会的プロセスそのものを含んでいることから、設計にはコンピュータエンンジニアのみならず社会学者、心理学者、認知科学者等の学際的な協力が必要となる。こうした観点から、1984年に米国において学際的な会議であるCSCW（Computer Supported Cooperative Work)と呼ばれるワークショップが米国の大規模共同研究組織であるMCCの主催の基に始めて開かれた[Greif]。このワークショップは、以

後２年ごとにACM （米国のコンピュータに関する学会）の主催で開催されるようになり年々参加者を増やしている。最近では1992年11月にカナダのトロントで開催され 650名の参加者を得ている。一方ヨーロッパにおいても同様な関心が高まり1989年にロンドンで第１回のECSCW （EuropianCSCW）が開かれ、以後２年ごとに開かれるようになった。また、最近では製品レベルのグループウェアの展示会であるGroupware' 92と呼ばれる催し物が米国で開催され話題となった。

本稿では、グループウェアを種類によって分類し、各々について現状と動向について解説する。なお、グループウェアはグループのためのウェア（ツール）といた漠然としたものであり、扱う範囲としては次のような分野を含んでいる。

Augumented Knowledge Workshop

Collaborative Computing

Computer Assisted Communication

Computer Conferencing

Computer Supported Cooperative Work (CSCW)

Computer Supported Group (CSG)

Coordination Technology

Decision Conference

Electric Meeting System (EMS)

Group Decision Support System (GDSS)

Group Process Support System

Intelligent Communication

Interpersonal Computing

Interfunctional Coordination

Flexible Interactive Technologies for Multiperson Task

Technological Support for Work Group Collaboration

Vertual Reality

Working Computing

1. グループウェアの分類

グループウェアは、その使われ方からしばしば時間的特性と空間的特性により分類される［Ishii-1］。時間的特性とは、グループワークの時間的制約条件であり会議や電話のようなリアルタイム（即時）に行われる作業を支援するものと、文書や手紙を介した非リアルタイム（非即時）のグループ作業を支援するものに分かれる。空間的特性には、グループのメンバーが一堂に会して行う作業を支援する直接対面型とメンバーが散らばった状態における作業を支援する遠隔分散型とがある。こうして分けたカテゴリーごとのシステムのタイプを図１－１に示す。一般に、直接対面する環境においてグループ活動を非即時で支援するツールはない。なお、こうした分け方の他にグループのサイズを含めて分類する方法や、支援モデルで分類する方法もある。

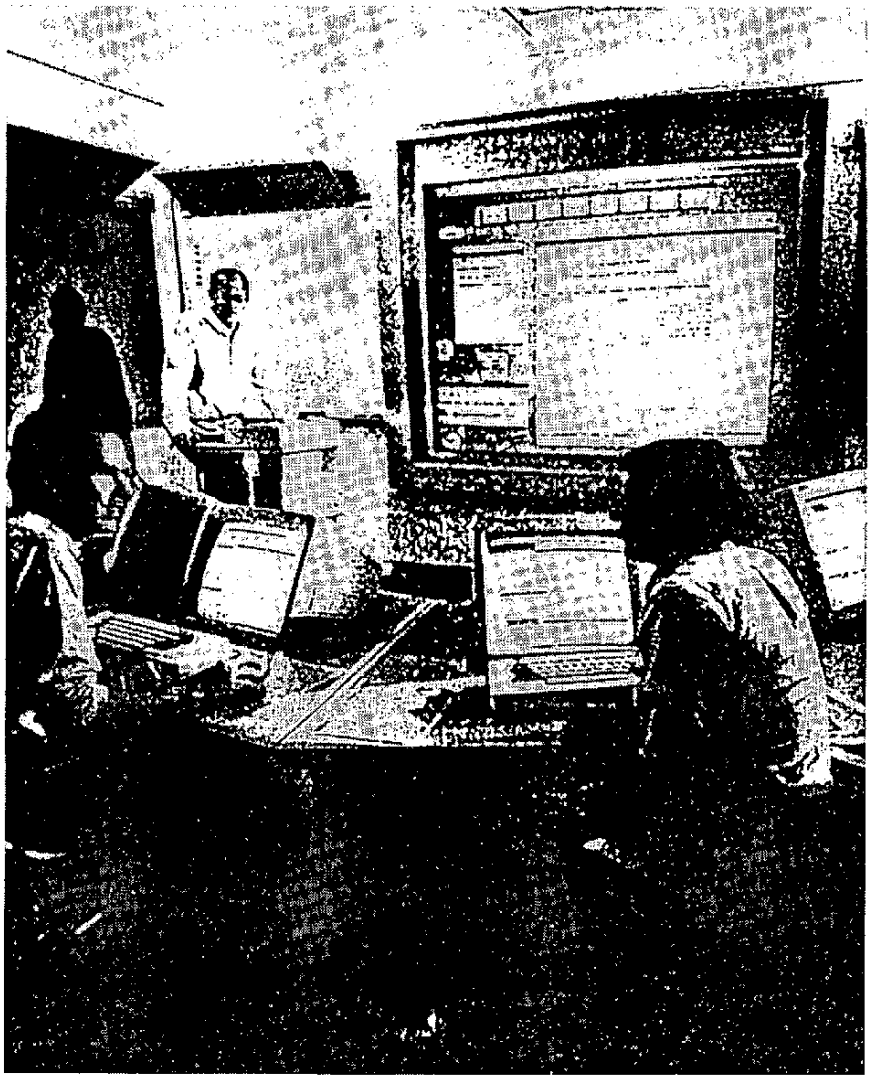
		空間的特性	
		直接対面型	遠隔分散型
時間的特性	即時型	●電子会議システム	●グループ専用エディタ ●ウインドウ共有／画面共有システム ●ワークステーション／ビデオ会議システム
	非即型		●ビジネス・コーディネーションシステム ●議論支援システム ●オフィスワークフローシステム ●メッセージ構造化システム ●協同文書作成支援システム

図1-1　グループウェアの種類

2. 即時直接対面型のグループウェア

このカテゴリーの代表は、電子会議システムである。電子会議システムというとインテリジェントビル等におけるテレビ会議システムを思い浮かべることができるがグループウェアでいう電子会議システムは、コンピュータベースのシステムを扱うものであり単にビデオ回線と音声回線で双方向の通信ができるだけのシステムは除いている。

最初の電子会議システムは、1967年にエンゲルバートがスタンフォード研究所で行ったオンライン会議まで遡ることができる。このオンライン会議室には、TSS システムに接続された6台の CRTディスプレイ装置が一つの会議室内に置かれた。各参加者は、同一の内容が映しだされる CRTディスプレイ装置を通してドキュメントの共有を行いながら会議を進めた。画面のスクロールなどのコントロールは、一人に限定されていたが、各参加者はマウスによりポインターを自由に動かすことが出来た（制御なし）[Engelbart]。当時のディスプレイ装置は、文字しか表示するしかできないなど使いにくいもので効果も限定されていた。しかし、最近に見られるコンピュータ技術の急激な向上は新しい電子会議システムをもたらした。その代表的なものとして XeroxPARCの Colabがある[Stefik]。 Colabは、２人～５人のグループの会議でのコンピュータ利用を目的に開発されたもので、Ethernetで結ばれた３台のXeroxワークステーションとLiveboardと呼ばれる大型スクリーンからなっている。各ワークステーションとL iveboardはビデオネットワークで結合され、お互いのスクリーンを見たり、 Liveboardに表示できる構成になっている。ColabにおいてLiveboardは、会議室における黒板の役割を果たしている。こうした実世界にあるものを真似てコンピュータシステム上に実現する方法をメタファという。従って、Liveboardは、黒板のメタファということになる。Colabの実験環境では、Cognoterと呼ばれるアイデアプロセシングシステムや Argunoterと呼ばれるプロポーザル評価システムが開発された。Cognoterについては、実験を通して設計時には予想もしなかった対話の障害が発見され、その改良版であるCnoterの開発が進められている[Tatar] 。



（左上にいる人はモデレータで会議の調整・進行を行う）

図1-2　Colab会議室（出典：[Stefix]）

富士ゼロックスでは、最近、海老名の研究所にColabと同等の環境を構成し実験を進めている。この他にも電子会議の実験システムとしてCapturelab(EDS)、ICICLE(Bellcore)、ProjectNICK(MCC)等が開発されている。

Capturelabは、市販のパーソナルコンピュータであるMacintosh をネットワークで結んだもので、通常のソフトウェアをそのまま使うことができ新たに開発する必要がない。各ユーザは同一の画面を自分用のパソコン上で見ることができるし、同じ内容が大画面にも映しだされる。またパワーキー（Machintoshの電源を入れるためのボタンで標準のキーボードについている）を押下することによりアクセス権を得ることができるよう工夫されている[Mantei-1]。

ICICLEは、知識ベース等を用いたC言語プログラムのインスペクション専用のシステムである。プログラムコードのインスペクション会議は、モデレータと呼ばれる調整役、コードを読み上げるリーダ、インスペクション結果を記述する書記、コード作成者及びインスペクタからなっている。インスペクション自体は各インスペクタが予め自分のワークステーションで行っておき会議に結果をもちよる。この場合、物理的にソースコードのドキュメントが配られるのではなく、ファイルをリモートアクセスするためのファイル名が与えられる。各インスペクタはファイルをリモートアクセスして検査を行う。このツールにより書記の作業が大幅に軽減されたとの報告がある[Sembugamoorthy]。

ProjectNICK は、打ち合わせ等の会議をポットの理論でモデル化しながらインプリメントしようとするものである[Cook]。

会議は、日常行われ馴染みの深いものであるが、その構造についてはほとんど知られていない。こうした実験環境において、会議の模様をビデオに収めたり、ネットワーク上のメッセージログを取り、それらを解析することにより会議の理解が進み、今後効果的なツールが開発されることが期待される。

なお、電子会議システムの特種な分野としてGDSS(Group Decision Support System) と呼ばれるものがある。GDSSは集団における意志決定を支援するシステムでありアリゾナ大学のPLEXSYS、ミネソタ大学のSAMMが有名である。PLEXSYSは、通常の電子会議システムと同じように1人1台のワークステーションと大型スクリーンから構成されている。各ワークステーションから入力された内容は、参加者全員が見ることのできる大型スクリーンに即座に写しだされる。PLEXSYS の会議手順は①予備会議室における会議の目的、形式に関する事前説明②問題解決に関するアイデアの匿名同時入力③イッシュアナライザーによる整理、再構成④匿名投票⑤選択された結果の蓄積のようになる[Kozai] 。

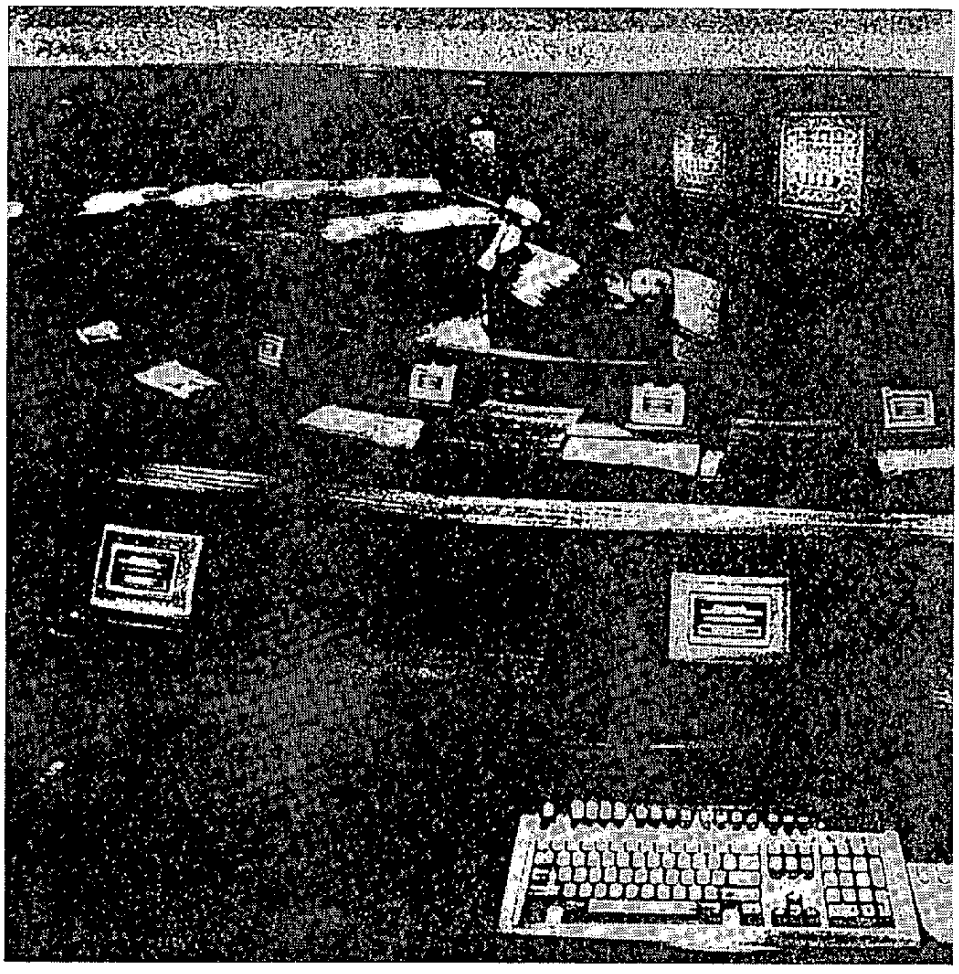


図1-3　アリゾナ大学のPLEXセンター（出典：[Valacich]）

グループ作業は、プロセズロスとプロセスゲインによって評価することができる。プロセスロスとは、会議において同時に一人しか発言できないといったプロダクションブロッキングを指す。

PLEXSYS では、各参加者に一台のパソコンが与えられ同時平行的に意見を入力することができプロダクションブロッキングを回避することができる。また、匿名入力、匿名投票によって意見の不平等な扱いの除去、特定の人による独裁、周囲からの圧力、独自性の低下が阻止できることから、意志決定が小数の人によって行われるといった感じが薄らぎ、連帯感を高める効果がある[Valacich]。

しかし、「グループを協調的にする技術を使って得られた判定は、グループの中の一番優れた意志決定者が単独で考えた判定より劣っていることがある」も事実でありこの辺

にGDSSの難しさがある[Kraemer]。

アリゾナ大学のシステムが最近IBMによりTeamFocus として製品化されるなど一部には市販化の動きもあるが、多くのシステムは実験段階に留まっている。

3. 即時遠隔分散型グループウェア

グループのメンバーが遠隔地に分散されている環境において、メンバー間での協調作業をリアルタイムで支援するものでLAN 、WAN 等のネットワークの利用が前提となる。また、この種のグループウェアでは、メンバーが分散されていることから、アクセス制御、作業の進行管理、同時実行制御をいかに行うかが重要な技術課題となる。

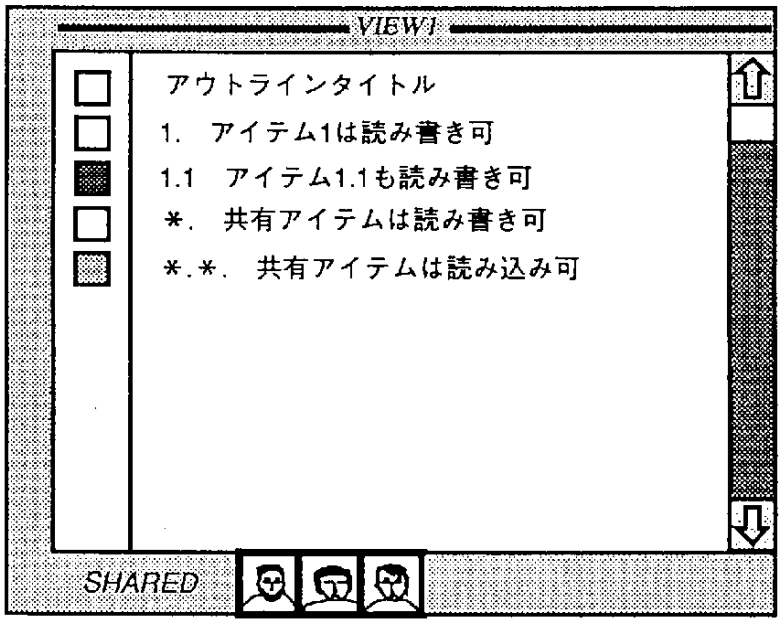
3.1 グループエディタ

テキスト文章あるいはグラフィックデータを、分散されているグループのメンバーのワークステーション上に表示し、各メンバーは自分のワークステーションでそれを見ながら同時に編集するシステムである。

MCC では、ほとんど何の制御もとらず、誰でも見ることができ、何でも編集でき、かつ編集中は何もロックされない、GROVE(the GRoup Outline Viewing Editor) と呼ばれるアウトライン編集エディタを開発し利用実験を行った[Ellis]。

GROVE を分散環境で利用したところ、各自が自分のオフィスに居たまま共同編集が行えることから自分のローカルな情報（本、資料等）へのアクセスが容易に行えた、グループ内の平行作業が増したとの良い評価を得た。しかし、議論を困難にし、グループの焦点を集めることが難しく、その結果疲れたとの悪い評価もあった。また、直接作業に入るため社交的対話が減少したとのどちらともとれる意見も報告されている。こうしたシステムを開発する場合同時アクセス制御をどうするかといた技術的な問題が話題になるがGROVE での実験では何の制御メカニズムを設けなくても人間の能力（例えば電話でも相手が喋ろうとしていることを会話の流れのなかで予期でき同時に喋ることはほとんどない）によって自然に同時アクセスが回避されたと報告されている。さらに、グループエディタに慣れるとシングルエディタの使用は苦痛になるという興味ある意見を得てい

る。同種のシステムとしては、この他にミシガン大学のShrEdit 、ヨーロッパEspritプロジェクトのCoAUTHOR等がある。



（参加しているユーザの顔がイラストで表示される。なお、他の例であるが、積極的にアクセスしている参加者の顔は活気のある顔にし、ほとんどアクセスしていない参加者の顔は寝ている顔にしているものもある。）

図1-4 GROVEの画面例（出典：[Ellis]）

市販されている代表的な製品としてGroup Technologies社のAspects がある[Group] 。Aspects には、ワープロ、ドローイングツーツ、ペインティングツールか組み込まれており、これらのツールを使って書いた文章や図のリアルタイムレビューが行える。各メンバーはネットワークで結ばれてさえいればどこにいてもかまわない。図１－５に画面例を示す。図の下方にあるウインドウでテレポインティングの種類を指定したり、アクセス権の要求等を行うことができる。一般にテレポインティングとは、一つの画面に複数のポインタを表示できるようにし画面上で各自が独自のポインタをもてるようにする技術をいう。この際、ポインタの形状が同じだと誰のポインタであるか識別できないため色で区別したり、形状を変えて区別している。

Aspects の画面は白黒であるため形状を変えられるようになっている。アクセス権の取得は、誰でも制限無くアクセスできる方式、アクセスしたいときに要求を出す方式、ア

クセスできる人が限定されている方式から選択できる。

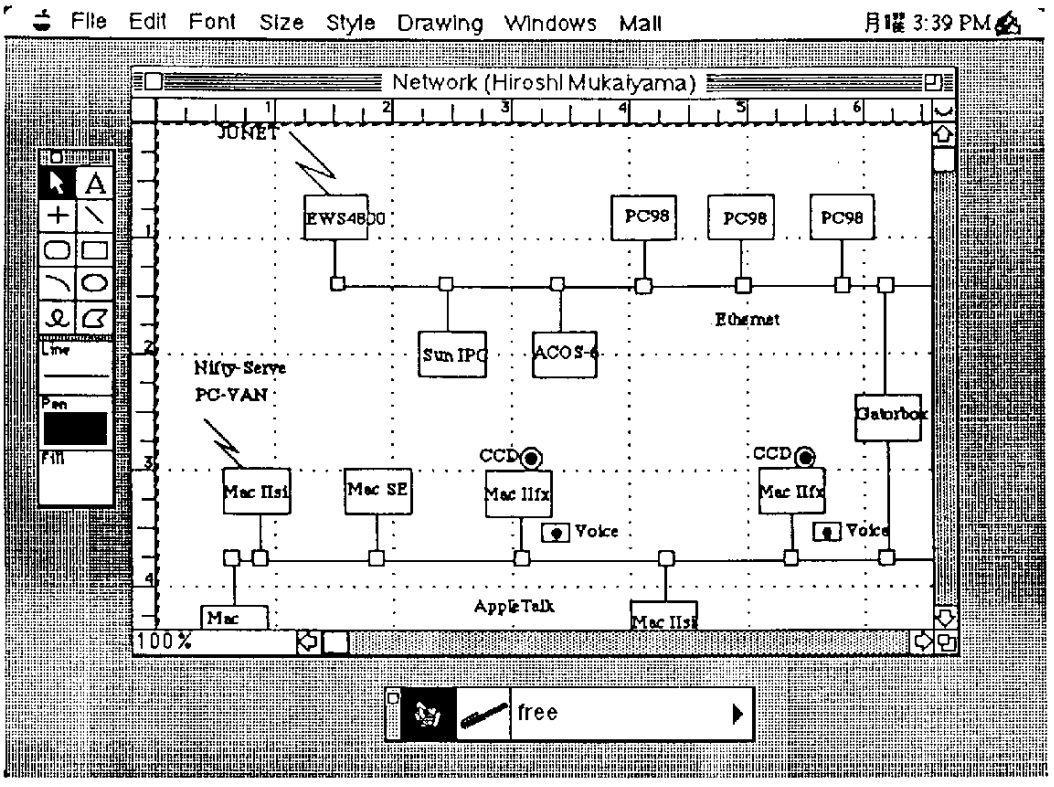


図1-5　Aspectsの画面例

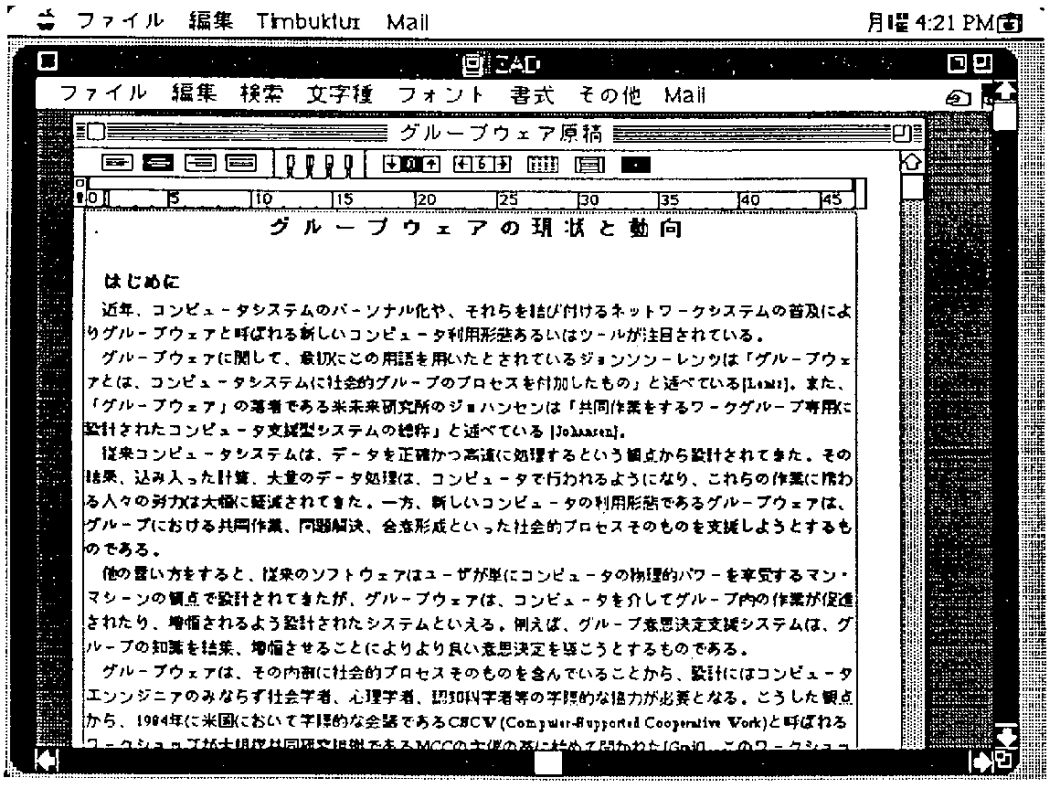
3.2　ウインドウ共有／画面共有システム

今まで述べてきたシステムは、いずれもワークステーション上のウインドウを共有したり、画面を共有してグループ作業環境を構築していた。この参加者が全員同じものを見ることができることを可能にさせる技術は、DTP(Desk Top Publishing)の世界で言われているWYSIWYG(What You See Is What You Get)を真似てXerox PARCのColabチームによりWYSIWIS(What You See Is What I See) と名付けられた。ここで述べるシステムもWYSIWIS を実現するものであるが、ウインドウや画面がアプリケーションと独立な点が異なる。

画面共有を簡易に実現したシステムとしてMacintosh 上で稼働するTimbuktsが有名である[Farallon]。Timbuktsでは、相手の画面全体を自画面内のウインドウに表示することが出来る。ただし、画面は縮小されないので小さなウインドウで見る場合は、スクロー

ルして見ることになる。アクセス制御としては、参照のみ、参照／編集とも可の２つを持っている。フロアパッシングの機能は持っていないので電話等の方法で制御することになる。

ワープロソフトウェアとTimbuktsを組み合わせて先のグループエディタと同様のシステムを構築できたり、アイデアジェネレーションソフトウェアとTimbuktsを組み合わせてColab のCognoterと同様なシステムを構築できるなど画面共有システムは、既存のソフトウェアとうまく組み合わせることにより簡易にシングルユーザ用ソフトウェアをグループ用ソフトウェアに変えることができる。また、離れているユーザの画面のポインターを用いてソフトウェアの操作法を教えたり、トラブルの状態を見ることができる。



（画面内に他の画面が取り込まれている）

図1-6　Timbuktuの画面例

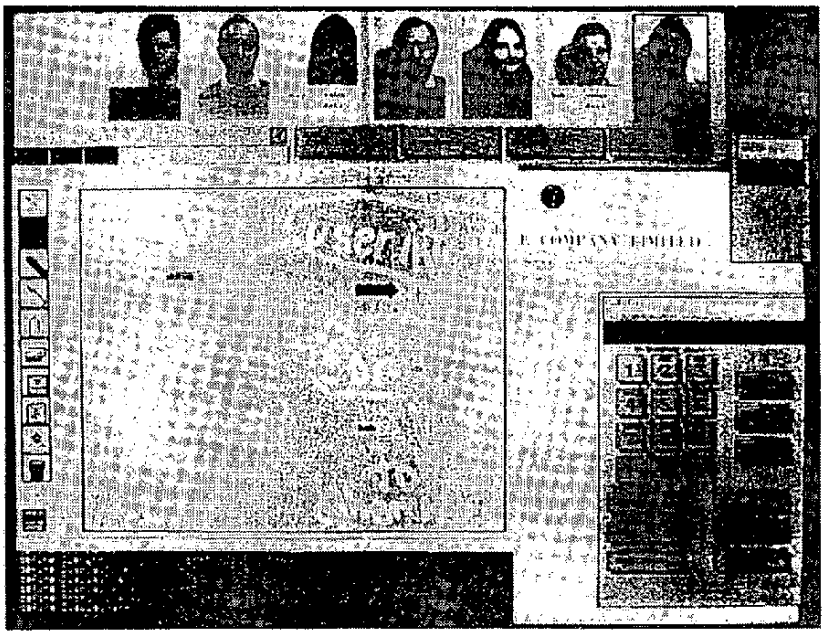
アプリケーションに依存しないウインドウ共有システムの実現方法としてアプリケーションとウインドウシステムの中間に会議エージェントモジュール（発言権管理、ワークスペース管理、動的再構成、秘書機能を行う）を持たせる方法が検討されている[Lauwers]。

3.3　ワークステーション／ビデオ会議システム

この種の多くのシステムは、テキスト、グラッフィク等のデータの他に音声や動画等の表現力が豊かなマルチメディアを扱っている。システムは大きく分けて、遠隔地に分散されている共同作業者に仮想会議室などの共同作業の場を提供するものと、遠隔地に分散されているメンバーにインフォーマルなコミュニケーションの場を提供するものがある。

前者の代表的なものとしてBellcoreのRAPPORT、NECのMERMAID、NTTのTeamWorkStationがある。RAPPORT は、ワークステーションと電話を使用して、音声、データ、イメージを共有しながら、リアルタイムで議論ができるマルチメディア会議システムである[Ahuja]。RAPPORT では、さまざまにシステム構成を変更し、パフォーマンス等に与える影響の実験が行われた。

最近では、参加者の顔をビデオカメラを通して動画として表示したり、ドキュメントをビデオカメラで映し出す機能が付け加えられている。



（画面上部のウインドウに参加者の写真が表示されている。中央左のウインドウは共有ウインドウで参加者全員のワークステーションに同じ内容が表示される。右下のウインドウには相手を呼び出すためのプッシュホンに似たボタンが表示されている）

図1-7　RAPPORTの画面例（出典：[Ahuja]）

MERMAID は、音声、データ、イメージの他に動画を扱えるようにしたマルチメディア会議システムである。音声、動画はメディアサーバに集められ合成された後、同報されている。動画は、最大４画面が合成され動画ウインドウに表示することができる。また、フロアパッシング制御としては、議長指名、要求順、バトン、フリーの４種類を持っている。日米６ケ所に設置した約２０台のWSによる試用実験に基づいて、①動画は、参加者に安心感を与え、参加者間の親近感を増大させた、②初対面の人が多い会議では、顔と音声の対応関係の把握が難しかった、③参加人数が多いと誰が発言しているか判別困難なことがあった等の結果が報告されている[JIPDEC]。

TeamWorkStationは、ビデオオーバレイの技術を用いてコンピュータ上のテキストとライブビデオ画像を合成、表示することができる。ライブビデオにペンを持った自分の手を写し、それを相手のコンピュータテキストと合成することによりあたかも相手のテキストに自分の手で上書きしているような効果が得られる[Ishii-2] 。

廊下でのちょっとした会話、オフィスにおける雑談のようなインフォーマルで非構造的な接触は、情報源として、あるいはグループのメンバーの親密感強化による円滑な共同作業といった点で極めて重要である。

物理的に分断された環境で、こうしたインフォーマルなインタラクションを仮想的に可能にさせようとしたシステムとしてXeroxPARCのMediaSpace、BellcoreのCRUISER、トロント大学のCAVECAT がある。MediaSpaceでは、パロアルトにある研究所とポートランドの研究所の間に２４時間使用可能なオーディオ／ビデオリンクを張りどのような使われ方がされるか等の実験が行われた[Goodman] 。両サイトには相似のコモンと呼ばれる部屋が設けられ、部屋の中にはVideo wallと呼ばれる大画面ディスプレイが置かれ、鏡に相手が映っている感覚で会話が行われた。その結果、こうしたオーディオ／ビデオリンクは、２つのサイトの間で文化を共有し、良好な関係を保持する上で役だったと報告されている。CRUISER は、オフィス内や廊下にビデオカメラを多数設置し、それらからの映像を次々と切り替え、ワークステーションから仮想的にオフィス内を動き回ることができるようにしたものである。オフィス内での動きにしたがって自然発生的に行われるインフォーマルな社会的インタラクションをソーシャルブラウジング(social browsing)と呼び、社会関係を形成し保持するうえで極めて重要であるとしている[Root]。CAVECATは、ワークステーションとそれらを結ぶデジタル・オーディオ・ビデオ回線から構成さ

れている。各ワークステーションはパソコン、TVカメラ、TVモニタ、スピーカ、マイクロフオンから構成され、TVモニタには最大４ケ所の映像を表示することができる。CAV-ECATでは、こうした環境を用いて小数の人がオフィスを離れることなく共同作業を行う際の、技術的な問題、心理的、社会的影響について研究している[Mantei-2]。

4. 非即時遠隔分散型グループウェア

オフィス内あるいは遠隔オフィスに分散されたメンバーが電子メール等を用いてメッセージの交換を行いながら協調的に作業を進めていくためのシステムである。対象としている作業が非同期であることから何らかの協調モデルを用いて支援を行っているものが多い。

4.1 ビジネス・コーディネーションシステム

ウイノグラードらは、言語行動（Speech Acts）理論に基づいて会話をモデル化し、The Coordinator と呼ばれる対話支援システムを開発した[Winograd-1]。会話モデルは、図１－８のように会話の流れを状態遷移で示したものである。図中、円は会話の状態を示し、円と円との間の線は、言語行動を示す。実際の会話は状態を次々に変化させながら進む。また各状態では、可能な言語行為が示されている。例えば、①でＡからＢに対して要求が出されると②の状態になり、ここでＢのとりうる言語行動は、約束、拒絶、逆提案であり、Ａは要求を取り下げることができる。②でＢが逆提案すると⑥の状態に移り、Ａは逆提案の受理、逆提案の逆提案、逆提案の拒絶という言語行動を取ることができる。通常は、状態②、状態③、状態④と進み、Ａが結果に満足したことを宣言して、会話が終了する。

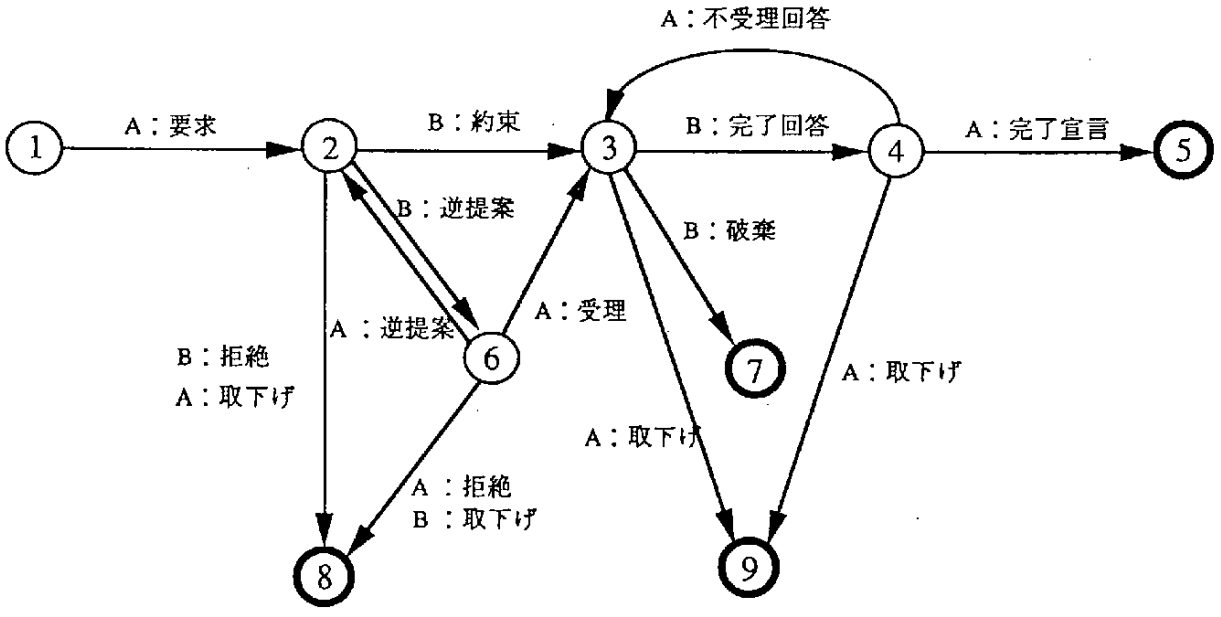


図1-8　会話のモデル（出典：[Winograd-1]）

The Coordinator では、こうした会話モデルに従ってメッセージ交換の支援機能を提供している[Winograd-2]。図１－９は初期画面でありここで、要求（Request)を選択すると要求を伝えるためのメッセージの型紙が表示される。型紙では宛先、カーボンコピーの送付先、ドメイン、カテゴリー、アクションを指定するエリアをもち、テキストを入力するエリアには、「あなたの要求は何ですか」というテキストが書かれている。ここに要求する内容を記述し相手に送る。このメールを受け取るとシステムは現在の会話遷移状態を基に図１－10に示す返事のための画面を表示する。画面の右側には約束（Promise）、逆提案（Counter-offer）、拒絶（Decline）と会話状態を変える行動が選択できるようになっている。左側は状態を変えない行動である。いずれかを選択すると対応したメッセージの型紙が表示される。The Coordinatorは更に会話の流れをトレースしたり、ペンディングとなっている案件を表示することもできる。The Coordinator は、効率的ではあるが、会話の形態を強制したり、厳格なトレースを行うことからナチ・ウェアとの批判もある。

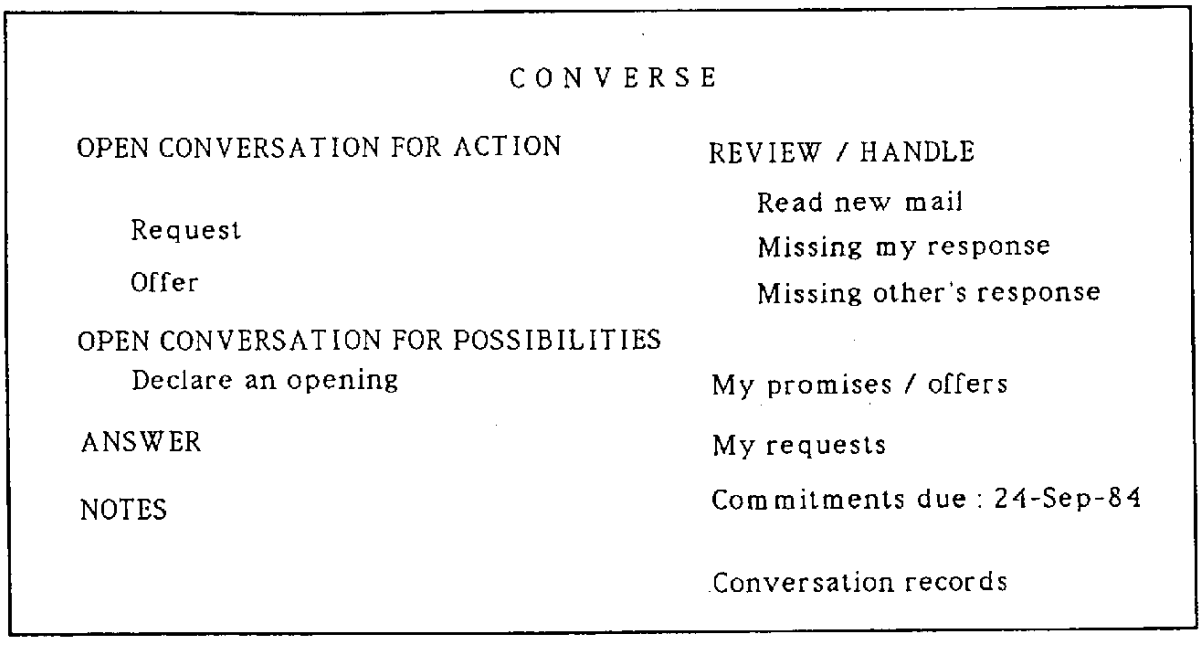


図1-9 The Coordinator初期画面（出典：[Winograd-2]）

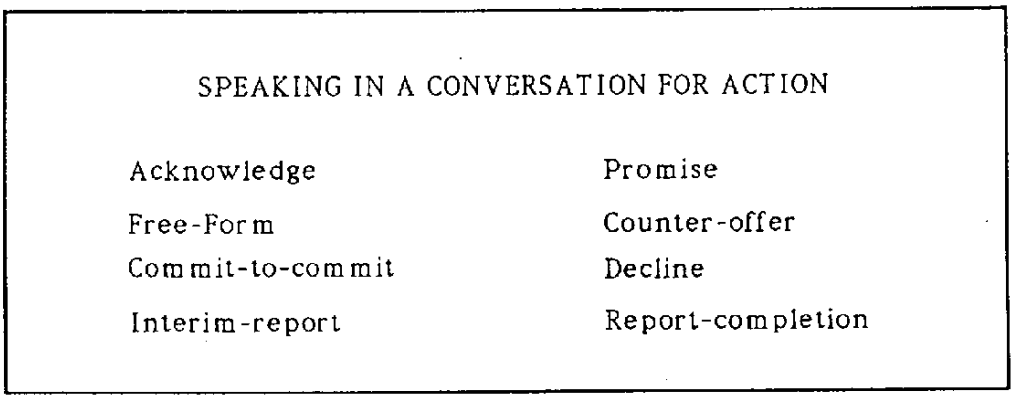


図1-10 返事のための画面（出典：[Winograd-2]）

ビジネス・コーディネーションシステムとしては、この他に個人のスケジュールデータベースを参照して会議の開催日時を調整するシステム等がある。

4.2 議論支援システム

MCCでは、gIBIS (graficIBIS) と呼ばれるハイパーテキストベースの議論支援システムを開発した[Conklin]。gIBISは、議論をIssue（問題）、Position（見解）、Argument

（意見）という枠組みで表現したIBIS(Issue Based Information System)モデルを基本にしている。例えば、CPUを何にするかといった問題に対して、A社のCPUにする、B社のCPU にするといった見解が示され、各々に対して、賛成、反対の意見が出されることになる。議論の過程を蓄積しておくことにより、議論の漏れや矛盾を回避できる。また、決定がどのような背景の基になされたかを知ることができる。こうした結論に至る過程の記録はオーガニゼイションメモリあるいはコーポレイトメモリと言われ最近注目されている。なお、gIBISは開発者のConklinにより商品化されCM/1として販売されている。

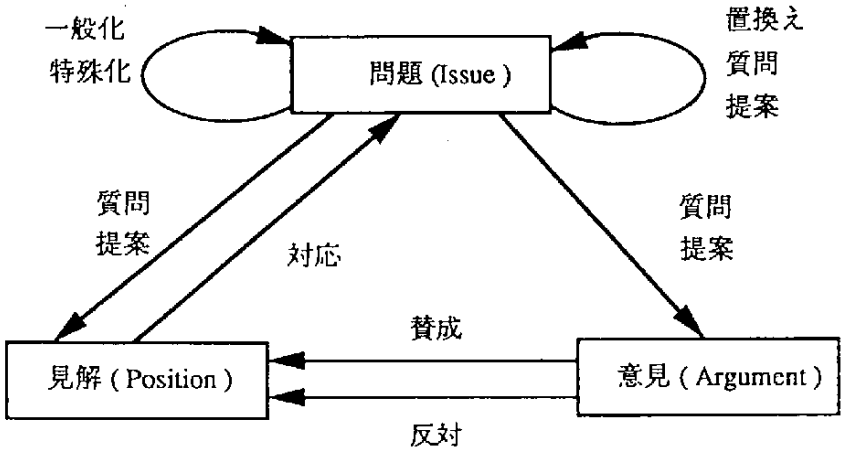


図1-11　IBISモデル（出典：[Conklin]）

MCC では、gIBIS の他に同じIBISモデルを用いてシステム設計手法をモデル化している。このモデルでは、成果物、設計ステップ、問題、見解、意見の５つの実体と８つの２項関係で構成されている。設計ステップは、成果物の生成や変更を行い、設計時の推論結果、根拠は問題点、見解、意見としてまとめられる[Potts] 。

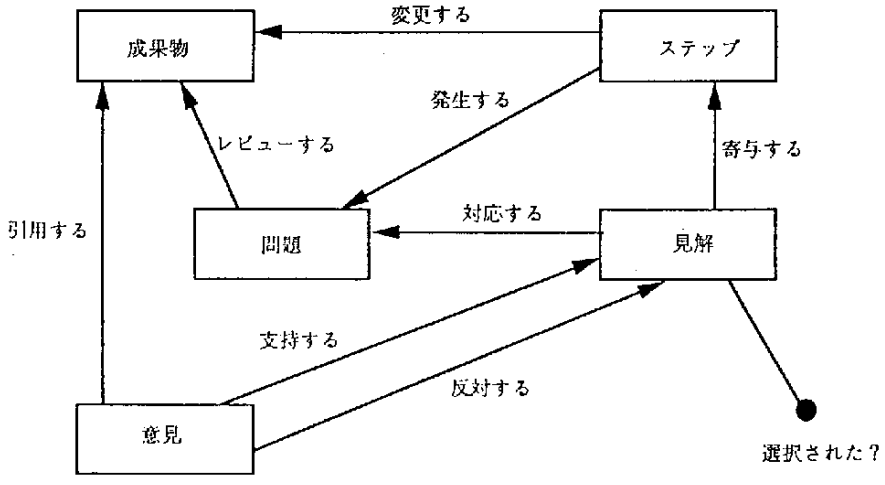


図1-12　Pottsのモデル（出典：[Potts]）

4.3　オフィスワークフローシステム

オフィスワークフローシステムは、オフィスにおける定型的なプロセスを手続きモデルとして構造化し、電子メールを用いて支援するシステムである。メッセージは、手続きモデルに従ってセントラルエージェントにより集中管理される。こうしたシステムの代表的なものとして、GMD （ドイツ国立情報研究所）のDOMINOがある[Kreifelts]。

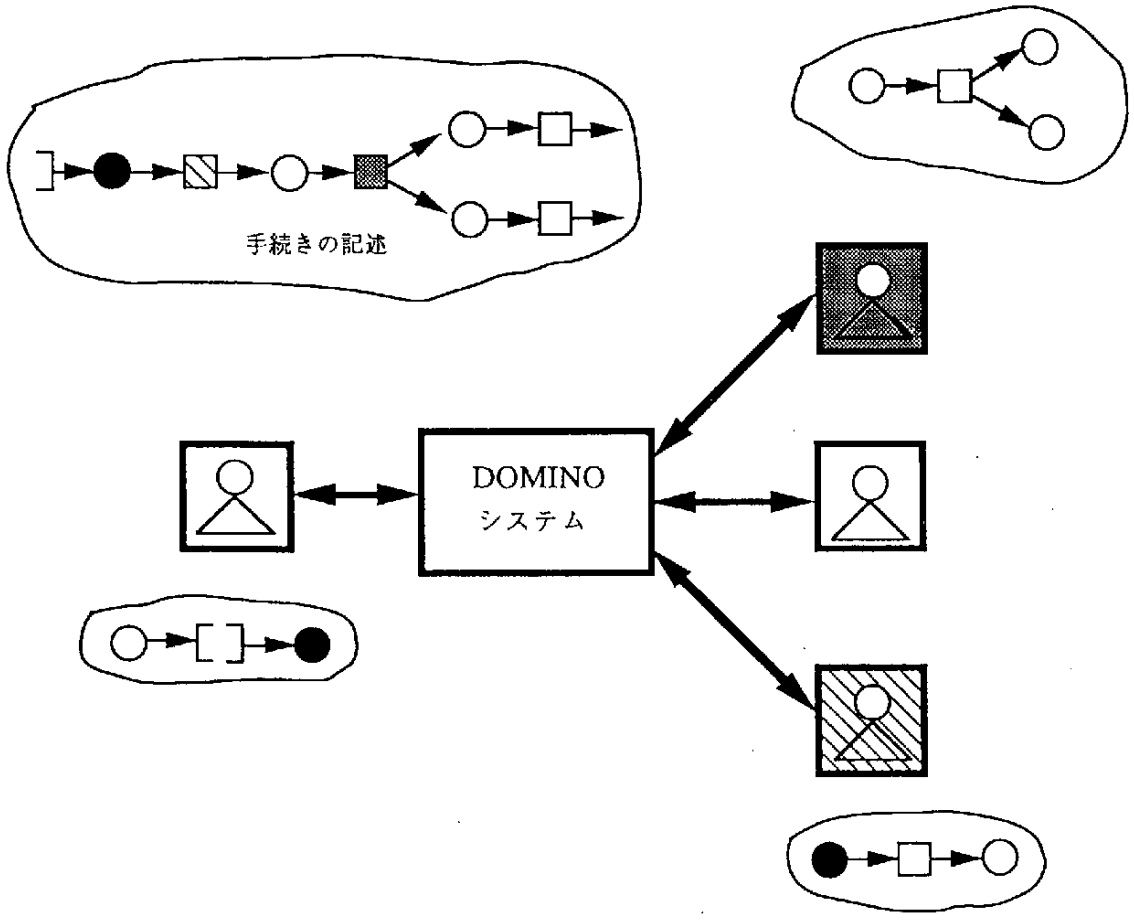


図1-13　DOMINOシステムの構成（出典：[Kreifelts]）

GMD では、DOMINOシステムをパソコン等の購入手続きに試用した。購入手続きは、購入希望者が、ワークステーション上で注文を入力することで始まる。この後、DOMINOシステムにより、この注文書はマネージャ、ディレクタと次々に渡され、必要なチェク等が

行われた後、購買部門に渡る。無論、途中で戻されることもある。購入手続きにDOMINOを３ケ月試用した結果、ユーザインタフェース、手続きがどこまで進んでいるかを示すトレーサビリディで良い評価を得たものの、例外処理や柔軟性に欠けている、組織・グループの概念がうまく表現できなかったとの指摘を受けている。

手順に柔軟性を持たせたものとしては、ESPRITのProMInanD におけるECF(Electronic Circulation Folders) がある[Karbe]。ECFは、オフィス内で循環されているフォルダ（書類）と書類受けを電子的に実現したものである。書類は、あらかじめ記述した移送手順によって転送されるが、作業者は必要に応じて作業ステップの追加を指示したり、拒否を行うことができる。作業ステップの追加によりあらかじめ定めてない仕事の実行も可能となる。

例外処理をどの程度認め、柔軟性をもったシステムにするかは、今後こうしたシステムを開発するうえでのポイントとなろう。

4.4　メッセージ構造化システム

先のオフィスワークフローシステムは、作業者間のメッセージを集中的に制御していたが、メッセージ構造化システムでは、そうした集中制御メカニズムを持たずに協調作業の支援を実現している。その代表的なものとしてInformation Lensがある[Malone-1]。

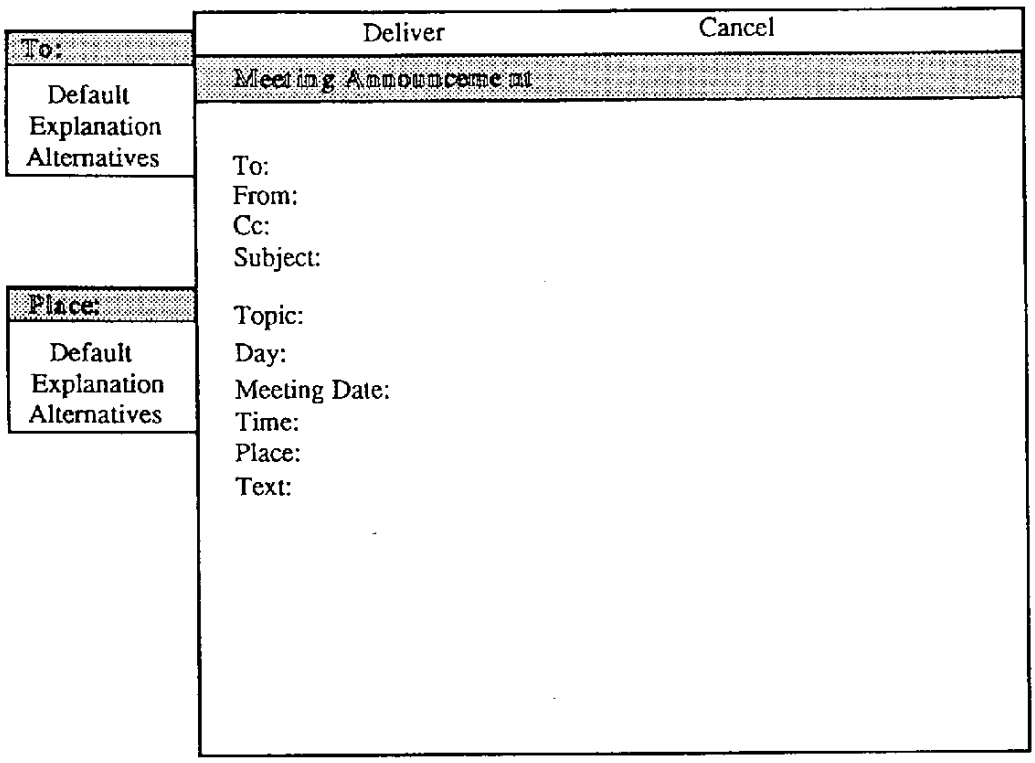


図1-14　メッセージ構造化テンプレート（出典：[Malone-1]）

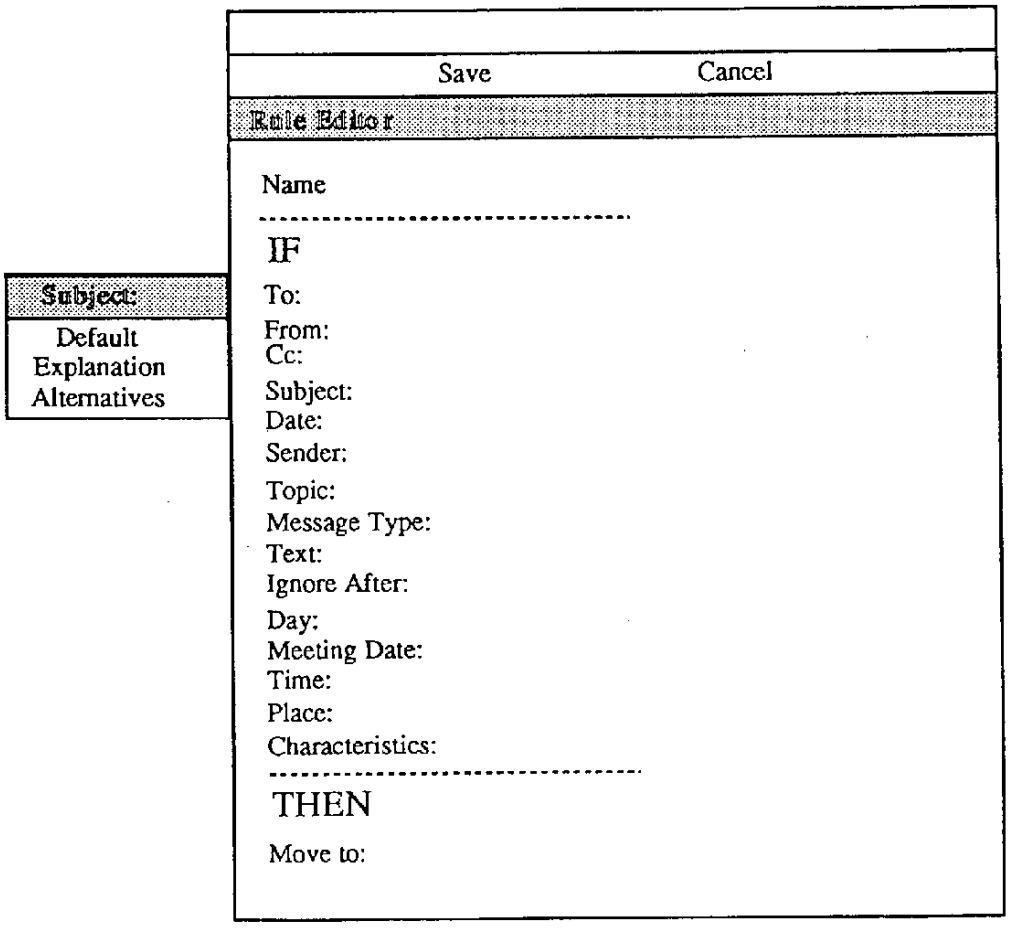


図1-15　ルールエディタ（出典：[Malone-1]）

Information Lensは、メッセージを準構造化することにより、知的処理を行おうとするものである。ここで、構造化とは、従来の自然文テキストイメージのメッセージを、いくつかのフィールドから成るメッセージにすることである。準構造化とは、すべの内容を完全にフィールド化するのではなく、可能な部分のみをフィールド化するという意味である。こうすることにより送信ユーザは、いくつかのフィールドから成るテンプレートを用いて穴埋的にメッセージを作ることができる。また、受信ユーザは、フィールドの値を条件に、メッセージの選択、移動等ができるようになる。更に、受信メッセージに応じて対応する返信メッセージの種類を示したり、自動的にアクションを起動することもできる。アクション起動では、例えばミーティングの通知がきたならば、カレンダーへの登録というアクションをとることができる。

Information Lensでは、テンプレート作成用のエディタ、受信したメッセージに対する処理をIF-THEN 形式で記述するルールエディタを持っている。これらエディタを用いて、グループの目的に合った準構造化電子メールシステムを構築することができる。

MIT ではInformation Lensを発展させたOvalと呼ばれる新しいツールの開発を進めいている。Ovalは、従来Object Lens と呼ばれていたものを改名したものであり、メッセージのオブジェクト化が計られている。即ち、新しいメッセージはオブジェクトベースのインスタンスとして生成でき、メッセージに対する操作が継承される[Malone-2]。

4.5 協同文書作成支援システム

電子メール、ファイルサーバ等を用いて非同期で文書の協同執筆、レビュー等を行うシステムでFor Comment、Quilt、PREP等がある。

For Comment では、サーバに蓄積したオリジナル文書に対して、各レビュアーがコメントを付けることができる。著者は、それらのコメントに従って文章の変更を行う。著者がアイデア募集といったコメントをあらかじめ付けておき、レビュアーがそれに答えるといった使い方もできる。

Quilt では、文書の種類及び共著者間の役割に応じて文書へのアクセスを制限することができる。文書に対する注釈は、自分宛ての注釈、読み手を指定した注釈、全員に対する注釈等細かく指定できる。その結果、文書作成中に「もう一度考えよう」といった自

分宛ての注釈をテキスト形式または音声で残したり、共著者を指定して意見を求めるといったことができる。また、文書に変更が生じた場合、関連する人に電子メールを用いて自動的に通知するトリガメカニズムや変更動作記録を自動的にとるアクティビティログの機能も有している[Leland]。

PREPでは、文書をチャンクとして細分化し、チャンクに対して、コメントを付けたり、コメントに対するコメント、ある文書に対する複数のコメント付けといったことが行える。また、PREPでは、電子媒体と紙媒体（ハードコピー）の融合を考えてPREPで紙をスキャンし、注釈を付けたのちプリントアウトして返すといった機能も検討している[Neuwirth]。

おわりに

Maloneは、サイバネティックスや認知科学と同様な学際的な研究分野として協調理論を挙げている[Malone-3]。

Maloneによれば協調プロセスは調整、グループ意志決定、通信、共通オブジェクト認識の４レベルの階層構造で構成される。

レベル	プロセス	システム例
調整	目標分解	Polymer
	タスク割当	Coordinator
	リソース割当	Information Lens
	同期	会議スケジュールツール
	順序付け	Polymer
グループ意思決定	代替案提示	gIBIS, Sibyl,電子ブレーンストーミングツール
	代替案評価	gIBIS
通信	メッセージ転送	電子メール
	ルート設定	Information Lens
共通オブジェクト認識	共有情報の参照と操作	電子会議、協同執筆ツール

図1-16　協調プロセスのレベル（出典：[Malone-3]）

層の順序は、各層が下の層のプロセスを利用するという考え方で構成されている。例えば、グループ意思決定では、メンバーが何らかの形で通信できなければならない。ただし、OSI 等のネットワークプロトコルと違ってある層が上の層のプロセスを利用する場合もある。例えば、通信するための共通言語を拡張するために意思決定プロセスを使ったり、意思決定アクティビティをアクターに割り当てるために調整プロセスを使うこともある。

今後、コンピュータサイエンス、経済学、組織理論、生物学等の分野でばらばらに行われていた協調に関する研究がこうした枠組みで新たに捕え直され協調理論として発展する可能性がある。また、そうした理論に基づいたアプリケーションが開発されてくるものと考えられる。

参考文献

[Ahuja] Ahuja, S. R. : A Comparison of Application Sharing Mechanisms in Real-Time Desktop Conferencing System, ACM SIGOIS Bulletin, Vol.11, No.2, pp.238-248, 1990.

[Conklin] Conklin, Jeff et.al. : gIBIS : A Hypertext Tool for Exploratory Policy Discussion, CSCW'88 Proceedings, pp.140-152, 1988.

[Cook] Cook, Peter et.al. : ProjectNick : Meetings Augumentation and Analysis, ACM Trans. of OIS, Vol.5, No.2, pp.132-146, 1987.

[Ellis] Ellis, Clarence : GROUPWARE : Some Issues and Experiences, Comm. of the ACM, Vol.34, No.1, pp.38-58, 1991.

[Engelbart] Engelbart, Douglas C. et.al. : A Research Center for Augumenting Human Intellect, AFIPS Conference Proceedings, Vol.33, FJCC, pp.395-410, 1968.

[Farallon] Timbukts User's Guide, Farallon Computing.

[Goodman] Goodman, George O. et.al. : Collaboration Research in SCL, CSCW'86 Proceedings, pp.246-251, 1986.

[Greif] Greif, Irene : Computer-Supported Cooperative Work : A Book of Readings, Morgan Kaufmann Publishers, 1988.

[Group] Aspects, Group Technologies, Inc., 1990.

[Ishii-1] 石井裕:コンピュータを用いたグループワーク支援の研究動向, コンピュータソフトウェア, 日本ソフトウェア科学会, Vol.8, No.2, pp.110-114, 1991.

[Ishii-2] 石井裕 : TeamWorkStation : Towards a Seamless Shared Workspace, CSCW'90 Proceedings, pp.13-26, 1990.

[JIPDEC] グループワーク支援システムの研究開発報告書, 日本情報処理開発協会, 1992.

[Johansen] Johansen, Robert : Groupware, The Free Press, 1988. (会津泉訳: グループウェア,日経BP,1990)

[Karbe] Karbe, B. et.al. : Support of Cooperative Work by Electronic Circulation Folders, ACM SIGOIS Bulletin, Vol.11, No.2, pp109-117, 1990.

[Kozai] 香西敏弘他 : アリゾナ大学のGDSS紹介, 情報処理学会研究報告, 90-DPS-44, pp.4.1-4.13, 1990.

[Kraemer] Kraemer, Kenneth L. et.al. : Computer-Based Systems for Cooperative Work and Group Decision Making, ACM Computing Surveys, Vol.20, No.2, pp.115-146, 1988.

[Kreifelts] Kreifelts, Thomas : Experiences with the Domino Office Procedure System, ECSCW'91 Proceedings, pp.117-130, 1991.

[Lauwers] Lauwers, J. Chris : Collaboration Awareness in Support of Collaboration Transparency : Requirements

for the Next Generation of Shared Window Systems, CHI'90 Proceedings, pp.303-311, 1990.

[Leland] Leland , Mary D. P. et.al. : Collaborative Document Production Using Quilt, CSCW'88 Proceedings, pp.206-215, 1988.

[Lentz] Johnson-Lentz, P. & T. : Computer-Mediated Communication, Academic Press, 1982.

[Malone-1] Malone, Thomas et.al. : Semistructured Message Are Surprisingly Useful for Computer-Supported Coordination, ACM Trans. of OIS, Vol. 5, No. 2, pp.115-131, 1987.

[Malone-2] Malone, Thomas et.al. : Experiments with Oval : A Radically Tailorable Tool for Cooperative Work, CSCW'92, pp.289-297, 1992.

[Malone-3] Malone, Thomas et.al. : Toward an Interdisciplinary Theory og Coordination, Center for Coordination Science MIT, 1992.

[Mantei-1] Mantei, Marilyn : Capturing the Capture Lab Concepts : A Case Study in the Design of Computer Supported Meeting Environments, CSCW'88, pp.257-270, 1988.

[Mantei-2] Mantei, Marilyn et.al. : Experiences in the Use of a Media Space, CHI'91 Proceedngs, pp.203-208, 1991.

[Neuwirth] Neuwirth, Christine M. : Issues in the Design of Computer Support for Co-authoring and Commenting, CSCW'90 Preceedings, pp.183-195, 1990.

[Potts] Potts, Colin : A Generic Model for Representing Design Methods, Proc. of the 11th International Conference on Software Engineering, IEEE, pp.217-226, 1989.

[Root] Root, Robert W. : Design of a Multi-Media Vehicle for Social Browsing, CSCW'88 Proceedings, pp.25-38, 1988.

[Sembugamoorthy] Sembugamoorthy, V. et.al. : ICICLE : Intelligent Code Inspection in a C Language Environment, COMPSAC, pp.146-154, 1990.

[Stefik] Stefik, Mark et.al : Beyond The Chalkboard: Computer Support For Collaboration And Problem Solving In Meetings, Comm. of the ACM, Vol.30, No.1, pp.32-47, 1987.

[Tatar] Tatar , Deborah G. : Design for Conversation : Lessons From Cognoter, Int J Man Machine Studies, Vol.34, No.2, pp.185-210, 1991.

[Valacich] Valacich, Joseph s. et.al. : Electric meeting support : the GroupSystems concept,Int J Man Machine Studies, 34(2), pp.262-282,1991

[Winograd-1] Winograd, Terry : A Language / Action Perspective on the Design of Cooperative Work, Human-Computer Interaction, Vol.3, pp.3-30,1988.

[Winograd-2] Winograd, Terry : A Language / Action Perspective on the Design of Cooperative Work, Human Computer Interacton 3(1), pp.3-30, 1988.

2章　グループウェアにおけるデータベースシステム構築技術

1.　データベースシステムの技術動向

1.1　データベースシステム

データベースシステム（DBS)とは、組織体（または、企業体）において、その活動を行うために必要となるデータを統合的に管理するためのシステムである。組織体の活動は、応用（プログラム）として、実現される。1960年代に確立された統合化の概念に基づいて、これまでに、ネットワーク型のデータベースシステムから、リレーショナル・データベースシステムが現在広く種々の分野で利用されてきている。データベースシステムは、図２－１に示すように、データベース、データベース管理システム（DBMS）、DD/D、応用プログラムから構成される。

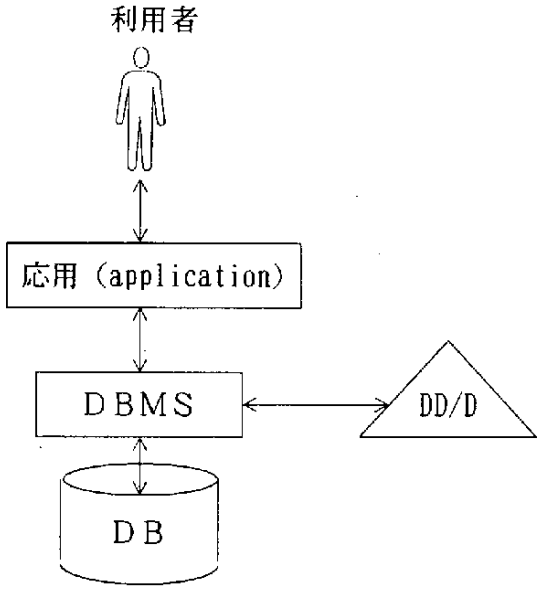


図2-1　データベースシステムの要素

1980年代から、ワークステーションと通信ネットワークの普及、発展により、情報システムの構成が変化してきている。これまで、大型コンピュータを中心とした集中型のシ

ステム形態から、種々のワークステーションがローカル・エリア・ネットワーク（LAN）で結合され、さらにこれらがネットワークにより相互結合（インタネットワーク）された分散型の形態となってきている。こうしたことにより、利用者は、種々のデータベースシステムを通信ネットワークを用いて利用してきている。データベースシステムも、複数のコンピュータ内にデータベースを分散させた形態や、データベース管理システム(DBMS)の機能をサーバとクライアントに分散させた機能分散の形態となってきている。

1.2 リレーショナルモデル

リレーショナルモデルは、IBM のE. F. Coddにより提案されたデータモデルであり、抽象代数に基づいた数学的なモデルである。現在、ISO により関係型言語SQL の国際標準化がすすみ、種々の応用で用いられてきている。今後のデータベースシステムで主に利用されていくものである。

1.2.1 リレーショナルモデル

リレーショナルモデルのデータ構造は、テーブルから構成される。例を図２－２に示す。これは、従業員の情報を示すテーブルEmp は、従業員の番号(eno)、名前(ename)、給料(sal)、所属している部の番号(dno) に関する情報を持っている。これらを属性またはカラムとする。テーブルDeptは、部についての番号(dno)、名前(dname) の情報を持っている。テーブル内の各情報、例えば、<1, A, 150, 1>を行とする。

Emp	eno	ename	sal	dno
	1	A	150	1
	2	B	200	1
	3	C	300	3
	4	A	100	2
	5	D	280	3
	6	E	450	1
	7	F	130	2
	8	D	310	2
	9	G	200	1
	10	H	280	1

Dept	dno	dname
	1	DDD
	2	EEE
	3	FFF
	4	GGG

図2-2 テーブルの例

1.2.2 操作演算

リレーショナルモデルの操作演算としては、SQL がある。SQL は、第一階言語に基づいた非手続き的な高水準言語であり、国際標準言語である。ここでは、SQL について概説する。

まず、図２－２に示したテーブルEmp (eno, ename, sal, dno)のスキーマは、以下のcreate文により定義できる。

```
create  table Emp
    ( eno  int NOT NULL,  /*属性enoのデータ型は整数で、空値が許されない*/
      ename  char(36),    sal    int,    dno    int  )
```

次に、テーブルEmp とDeptに対するSQL による検索と更新の例を説明する。

(1) DDD 部員のなかで、給料が 200以上の者とその給料を求めよ。

```
select   ename, sal   from  Emp, Dept
where    dname = "DDD"  and  Dept.dno = Emp.dno  and
         sal >= 200
```

また, SQL では以下のようにも書ける。inの右側にはselect... という検索式が書かれている。このような検索式を入れ子形式という。このinは英語の関係代名詞を示し、dnoを修飾している。

```
select   ename, sal  from  Emp
where    sal >= 200  and
         dno  in  ( select  dno  from Dept
                    where  dname = "DDD")
```

(2) DDD 部員のなかで、給料が 200以上の者とその給料を、給料の昇順に求めよ。

```
select   ename, sal  from  Emp
where    sal >= 200  and
         dno  in  ( select  dno  from Dept  where  dname = "DDD")
order  by  sal
```

(3) 部毎の平均給料を求めよ。

```
select  dno, avg(sal)  from    Emp
group   by  dno
```

(4) DDD 部員の給料を 1.5倍にする。

```
update  Emp   set  sal = sal * 1.5
where dno in  ( select  dno  from Dept where   dname = "DDD" )
```

(5) DDD 部員を消去する。

```
delete  from   Emp
where  dno  in ( slelect  dno from  Dept  where  dname = "DDD");
```

(6) 組<11, K, 210, 1 >を関係Emp に追加する。

```
insert into   Emp ( eno, ename, sal, dno)
values     ( 11,  K, 210, 1)
```

(7) DDD 部員として、Kを追加する。

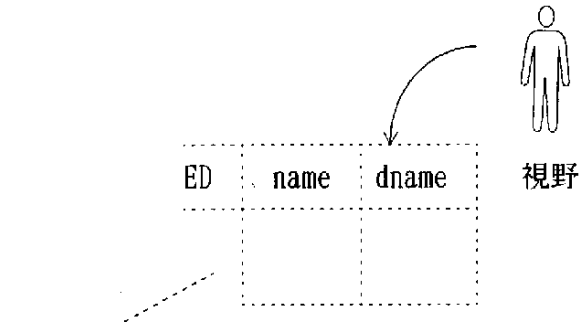
```
insert into  Emp ( eno, ename, sal, dno)
    select  11, "k", 210, dno   from  Dept
    where dname = "DDD"
```

(8) 図２－２に示したテーブルEmp とDeptの上に、従業員名と彼の属している部名を与える視野は以下のように定義できる［図２－３］。

```
create  view  ED ( name, dname)
as  select  ename, dname  from  Emp, Dept
    where  Emp.dno = Dept.dno
```

検索演算「DDD 部の部員を求めよ」は、次のように書ける。

```
select  name  from  ED  where  dname = "DDD";
```



Emp	eno	ename	sal	dno
	1	A	150	1
	2	B	200	1

Dept	dno	dname
	1	DDD
	2	EEE

図2-3　視野

1.2.3 まとめ

リレーショナルモデルは、以下の特徴を持っている。

(1) 簡単なテーブル形式をしたデータ独立なデータ構造を持つ。

(2) 非手続き的データ操作言語SQL を持つ。

(3) 閉包性がある。データベース内のテーブルら導出された結果はテーブルであり、再びデータ操作言語で操作できる。この性質は、データ構造を柔軟に変更できることを意味している。

(4) リレーショナルデータベースシステムは、中程度の性能をしている。リレーショナルデータベースシステムが、非手続き的な操作演算から物理的な操作演算の手続きを生成する。

(5) CAD 、個人用といった非定形的な業務に適する。

リレーショナルデータベース管理システムの例としては、以下のものがある。

(1) メインフレーム　　DB/2, SQL/DS(IBM)

(2) ワークステーション　　Oracle, Sybase, Informics, Empress, Unify

(3) パソコン　　Unify, Informics

2. グループウェアデータベースシステムの基本技術

グループワークデータベースシステムを実現する為の基本技術としては、オブジェクト指向データベースシステムとデータベースシステムの自律性と協調が重要となる分散型データベースシステムとがある。

以下に、これらについて解説する。

2.1 オブジェクト指向データベースシステム

2.1.1 基本概念

まず、オブジェクト指向システムの基本的な概念について考える。オブジェクト指向システムは、以下の基本的な概念に基づいている。

(1) オブジェクト

(2) メッセージ・パッシング

(3) 階層構造（汎化、集約化）

(4) 継承

(5) ポリモルフィズム

A. オブジェクト

オブジェクト指向システムの基本となるオブジェクトは、以下が一体化した概念である。

(1) 内部状態（データ構造）

(2) これに対する操作演算（メソッド(method)）

メソッドとは、オブジェクトが提供する操作演算である。利用者は、メソッドを用いてのみオブジェクトを操作できる。この意味で、オブジェクトは、メソッドを抽象演算とした抽象データ型である。例として、オブジェクト『銀行』は、メソッドとして、入金、出金、照会といったものを提供している［図２－４］。利用者は、銀行オブジェクトに対して、メソッド入金を用いて、ある口座に入金を行える。

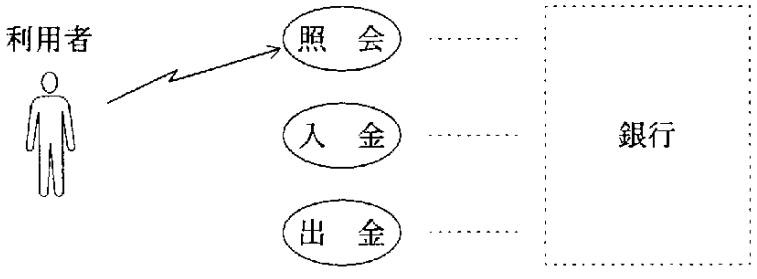


図2-4　オブジェクトの例

B. メッセージ・パッシング

メッセージ・パッシングは、システムの起動の方法に関している。従来のソフトウェアシステムでは、プログラムは、関数呼び出しにより他のプログラムを起動している。即ち、プログラムの制御が関数呼び出しの命令に達したときに、関数が呼び出され実行される。これに対して、オブジェクト指向システムでは、オブジェクトは、メッセージ・パッシングと呼ばれる機構により、活性化される。オブジェクト間では、メッセージと

いう通信単位の送受信により情報の交換が行われる。オブジェクトの動作を以下に示す。

［オブジェクトの動作］

(1) オブジェクトは、メッセージを受信したとき、活性化される。

(2) メッセージは、オブジェクトの持つメソッドを示すセレクタを含む。

(3) オブジェクトは、内部状態を持ち、状態により受け付けられるメッセージを受信し、セレクタが示すメソッドを実行し、内部状態を変化させる。

こうしたメッセージ・パッシングを実現する方法として、動的結合(dynamic binding)がある。すなわち、オブジェクトがメッセージを受信したとき、セレクタをこれが示すメソッドに結合するものである。従来のプログラム言語システムでは、コンパイル／リンク時に、全ての呼び出される関数は結合されている静的結合方式が用いられている。動的結合では、プログラムの実行時に、呼び出される関数が見つけられ、結合される。

C．階層化

複雑なシステムを構築したり、理解するためには、オブジェクトを階層化することが必要である。階層化の方法としてまず、抽象化がある。オブジェクトは、抽象化の概念により階層化される。

(1) クラスと例

同一の性質を持つオブジェクトの集合をクラスとする。クラスに属するオブジェクトをクラスのインスタンスとする。クラスとそのクラスに属するインスタンスオブジェクトの関係をinstance-of という。図２－５に学生のクラスとそのインスタンスの関係を示す。

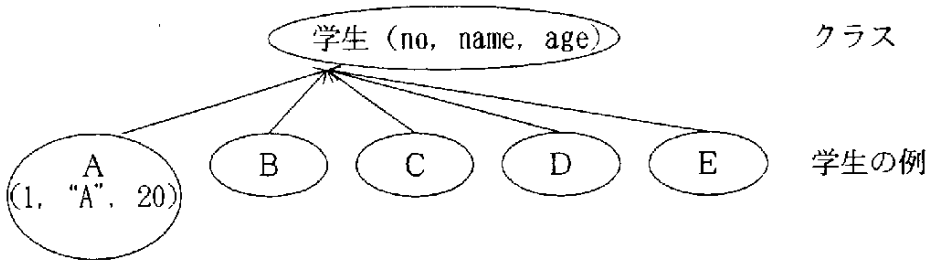
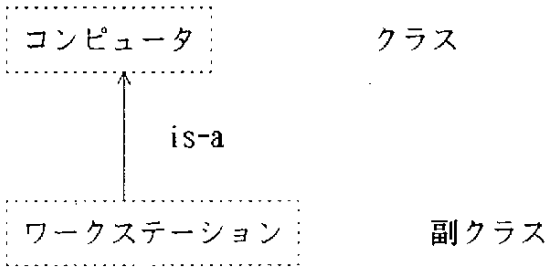


図2-5 クラスとインスタンスの関係

(2) 汎化(generalization)

クラス間の関係として、汎化（またはis-a）の関係がある。一つ以上のクラスから、共通の性質を抽象化して、より一般的なクラスを定義することが汎化である。抽象化された上位の概念と、より具体化された下位の概念の間の関係である。下位のクラス（class)に対して、上位を上位クラス（super-class)という。又、上位クラスに対して、下位のクラスを副クラス（subclass）ともいう。下図に上位クラスのコンピュータと下位のクラスのワークステーションの関係を示す。



(3) 集積化（aggregation)

集積化は、part-of といわれる関係である。あるクラスが、他のクラスから構成されていることを示している。図２－６にコンピュータが、CPU 、メモリ、ディスク、IO、バスから構成されるpart-of の関係を示す。

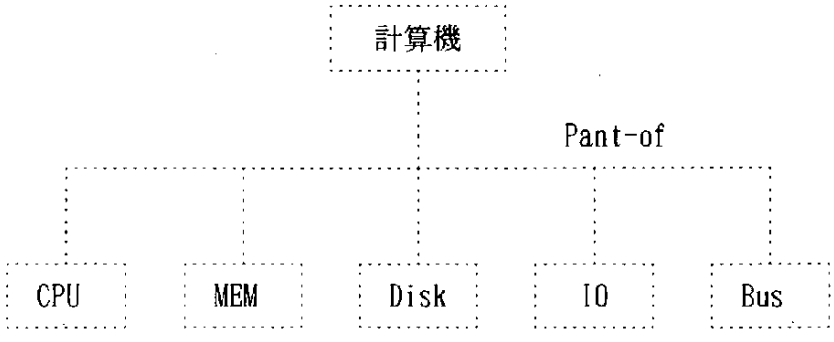


図2-6　集積関係

D. 継承

一般化では、上位のクラスの持つ性質、即ち、データ構造とメソッドが下位のクラスに継承される。図２－７に車の例を示す。

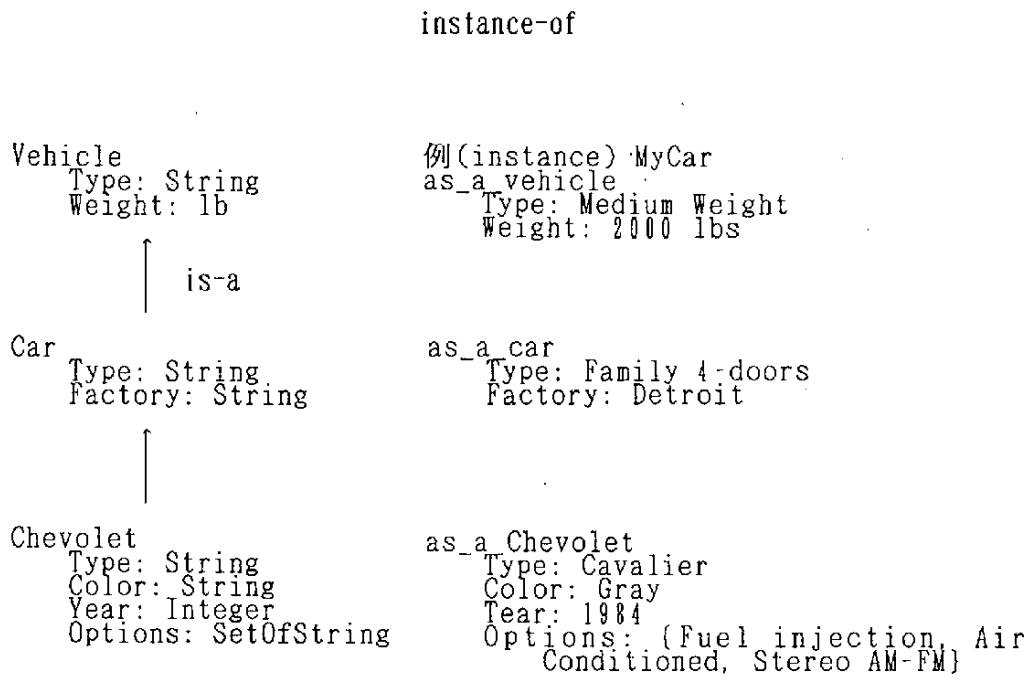


図2-7　継承階層

あるクラスが複数の上位のクラス$C_1,\ldots,C_n$を持ち、各上位クラスC_iから性質p_iを継承する場合を多重継承とする。多重継承では、継承される性質が競合する（conflict)場合が問題となる。例えば、図２－８で、勤労学生Student Workerは、Emp とStudentの両方の性質が継承される。ここで、Emp のbonus は給料に関していて、Studentのbonus は成績に関している。これは、継承する性質が競合する例である。競合の解消方法としては、上位のクラスに優先度を与えて、優先度の高いクラスの性質が継承されるものとする方法がとられている。この競合を、例外を示すために利用することも行われている。クラス間の汎化の関係が木構造をしているときは、多重継承は生じない。何故ならば、各クラスCは複数の上位クラスを持たないからである。これに対して、束である場合には、各クラスは一般に一つ以上の上位クラスを持てるので、多重継承の問題が生じる。

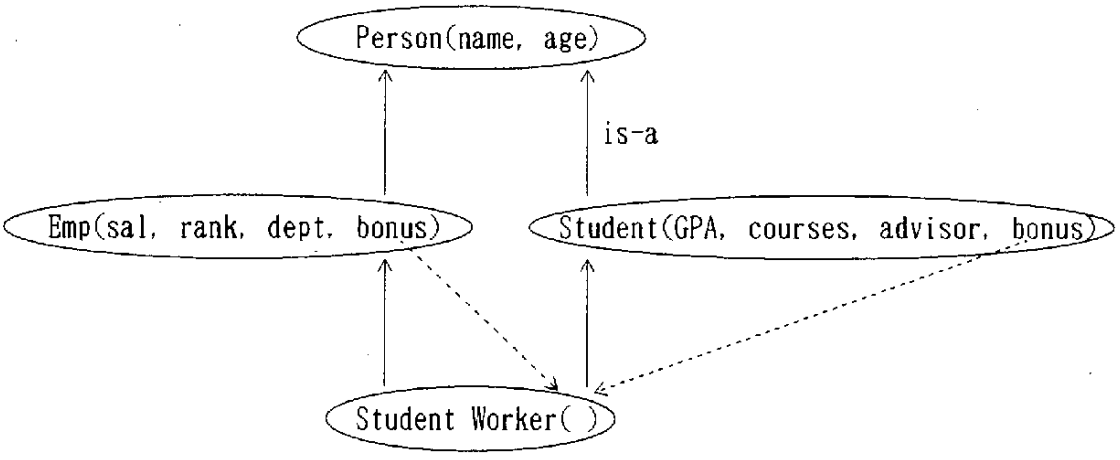


図2-8 多重継承

2.1.2 オブジェクト指向プログラミング・システムの例

A. Smalltalk-80

Smalltalk-80は、Xerox 社のPARCのソフトウェア概念グループが15年の期間をかけて開発したシステムである。「すべての人間の創造的な活動を計算機により支援すること」が目的である。Smalltalk-80は、プログラミング言語とプログラミング環境を統合したシステムである。Starワークステーション、InterLisp-D 知識エンジン(Xerox) 、Lisa, Macintosh (Apple社) に影響を与えた。OA、CAD 、電子出版等に影響を与え、高解像度ディスプレイ、アイコンを用いた利用者インタフェース等、現在のワークステーションの基礎を生み出している。Smalltalk の開発概念としては、以下の 3 点がある。

(1) パーソナル計算　　個人専用のワークステーション
(2) 対話　　視覚に基づいたグラフィックインタフェース
(3) オブジェクト指向プログラミング

Smalltalk のプログラムは、メッセージ式から構成される。例えば、通常のプログラム言語でのsin(theta)は、

theta sin

と単項メッセージ式により書かれる。これは、theta オブジェクトにメソッドsin を送信する (sin ⇨theta)ことを示している。また、3 と 4 の加算は、

3 + 4

と書かれる。この意味は、オブジェクト3にメソッド＋4を送信することである。このメッセージを二項メッセージ式という。この他に、以下のように、引数を持つ形式のキーワードメッセージ式もある。

```
householdFinances  spendAmount:560.00  forReason:'Rent'
```

B. C++

ベル研究所で、以下の目的のもとで、C言語に以下のオブジェクト指向機能を付加した発展型言語である。

(1) C言語固有のポータビリティの維持

(2) C＋＋とCの互換性

(3) 型チェックの強化（C言語では、型のチェックが弱い）。

(4) データ隠蔽

クラスは、Cのstructを一般化したclass 文により定義される。データ構造とともにこれを操作するための関数が与えられる。

```
class   date {
    int   day, month, year;
public:
    friend void setDate(date*, int, int, int);
    friend void nextDate(date*);
    friend void nextToday();
    friend void printDate(date*);
};
```

dateの実現方法を隠蔽し、4つのフレンド関数によってのみdateをアクセスさせる。dateの利用方法の例を以下に示す。

```
date   myBday, today;

setDate(&myBday, 9, 12, 1950);
setDate(&today, 21, 7, 1989);
printDate(&myBday);   nextDate(&today);
```

クラスの定義

```
class   date {
    int   day, month, year;
public:
    void set(int, int, int);
    void next();
    void print();
};
```

メンバ関数の参照

```
date   myBday;
...
myBday.print();
```

メンバ関数の定義

```
void  date.next(){
    day = day + 1;
    .....}
```

[例]

```
class  shape {
    vector  centrt;  int  color;  shape *next;
public:
    void   move(vector);
    vector where();
    virtual void  rotate(int);
    virtual void  draw();
```

副クラス　virtual 関数は、副クラス毎にオーバライドされねばならない。

```
class  circle : public  shape {
    float  radius;
public:
    void  rotate (int) {...}
    void  draw();
    .....
}
```

2.1.3　リレーショナルモデルでのオブジェクト指向

以前に概説したリレーショナルモデルは、集合論に基づいて構成されている。しかし、このような単純なモデルでは、CAD 、OA、知識ベースといった非標準な応用を処理できない。このために、集合指向のリレーショナルモデルに、オブジェクト指向の概念を取り組むことが試みられている。

A．リレーショナルモデルでのオブジェクト概念

A．原子的オブジェクト

原子的オブジェクト(atomic object) とは、上位の階層により原子的に扱われるデータ単位である。例としては、実数型、ドキュメント型、イメージ型がある。即ち、利用者は、原子的オブジェクトの内部構造とは独立に、これを利用できるものである。

既存のリレーショナルデータベース管理システム(RDBMS) は、整数型、文字列型といった組み込み型の限定されたデータ型だけを提供しているだけである。これに対する一つの解は、より強力な型を提供することである。例えば、配列、マトリクス、リスト、ビットマップ、イメージといったものがある。型には、単にデータ構造だけでなく、これに対する操作演算が与えられねばならない。これは、抽象データ型(ADT) の概念である。マルチメディアデータベースシステムを提供するためには、イメージ、音声、画像といった型を提供することである。

B．複合オブジェクト

複合オブジェクト(complex object)は、その属性が原子的でなくてもよく、階層的な構造を持つオブジェクトである。オブジェクトに含まれる非原子的なオブジェクトを、副

オブジェクト(subobject) 又は要素オブジェクト(component object)という。複合オブジェクトの例としては、OAでの、キャビネット等のOAオブジェクトがある。CAD でも、CAD 設計オブジェクトがある。例として、キャビネットオブジェクトのテーブル表現を考える。

```
Cabinet (cab#, properties,...)
Drawer  (drw#, cab#, properties,...)
Folder  (fdr#, drw#, properties,...)
Document(doc#, fdr#, description, date, author, ...)
```

リレーショナルモデルでは、テーブルは、正規化されているために、一つのオブジェクトが他のオブジェクトから構成されるという関係を一つのテーブルとして示せない。このために、ここで示したように、4つのテーブルが必要となる。あるキャビネット内の書類を操作するためには、これらのテーブルを結合する必要があり、操作が複雑となる。また、テーブル間のキャビネットとしての関係をとらえることも容易でない。複合オブジェクトを扱うためには、以下に述べる非正規形のテーブルが必要となる。

複合オブジェクト(complex object)概念を提供するデータモデルについて述べる。

［定義］オブジェクトは、以下のように再帰的に定義される。

(1) 整数、実数、論理値、文字列を原子的オブジェクトとし、これらはオブジェクトである。

(2) $O_1, \ldots, O_n$ をオブジェクトとし、$a_1, \ldots, a_n$ を属性とする。このとき、$<a_1: O_1, \ldots, a_m: O_n>$ を組オブジェクトとし、これはオブジェクトである。

(3) $O_1, \ldots, O_n$ をオブジェクトとしたとき、$\{O_1, \ldots, O_n\}$ を集合オブジェクトとし、これはオブジェクトである。

［例］

原子的オブジェクト	10
集合	{1, 2, 3, 4}
組	<name:Smith, age:42>
関係	{<name:Smith, age:42>, <name:Tom, age:30>}

階層組　　<name:<first:John, last:Doe>, age:40>

集合値を持つ組　　<name:Smith, children:{Jim, Tom}, age:42>

入れ子関係(nested relation)　<wine#:210, on_hand:{<store:10, qty:250>,
<store:20, qty:500>}> □

複合オブジェクトを示すテーブルについて考える。ワインオブジェクトについての情報を示す非正規形のテーブルWINEを図２－９に示す。

WINE

w#	on_hand		vinyard	vintage	price	
	store	qty			year	amt
210	20	510	Chenas	1982	83	5.00
	35	600			84	6.00
					85	6.00
320	10	500	Chablis	1981	85	15.00
	20	450				

図2-9　複合オブジェクトを示す非正規形テーブル

図２－９のテーブルの操作について考える。WINEテーブルから、vineyard＝"Chenas"で、amt ＜ 6.00なるものを求めることは、以下のように書ける。

```
select * from WINE
where  vineyard = "Chenas" and  price*amt < 6.00
```

ここで、price*amt は、price の中に、集合としてamt が含まれていることを示す。このSQL の結果は図２－10のようになる。

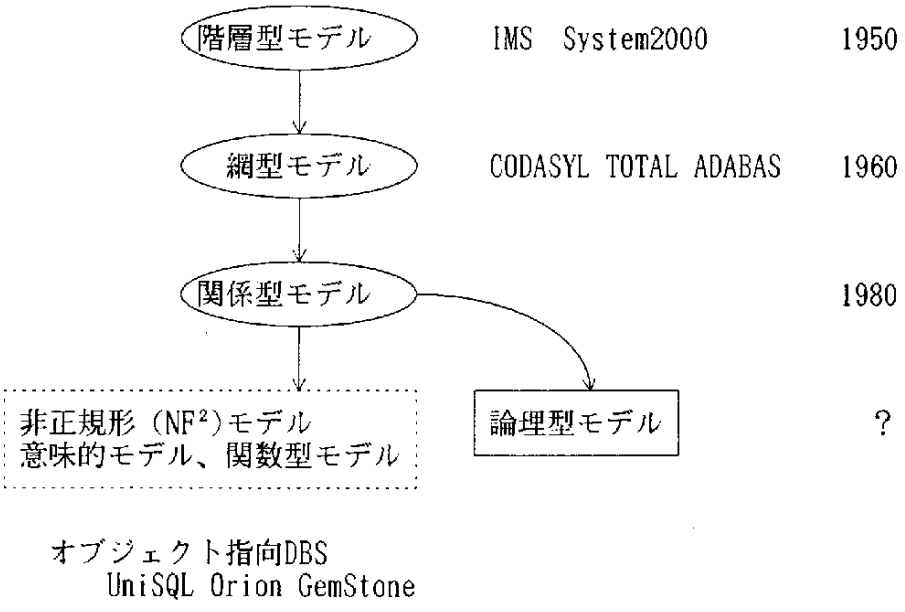
w#	on hand		vinyard	vintage	price	
	store	qty			year	amt
210	20 35	510 600	Chenas	1982	83	5.0

図2-10　結果

2.1.4　オブジェクト指向データベースシステム利用の効用

A．データモデルの発展

データモデルの発展を以下の図２－11に示す。



2-11　データモデルの発展

B．オブジェクト指向データベースシステム

データベースシステムの最小機能として、以下の点をあげることができる。

(1)　高水準質問言語と質問最適化　例えば、SQL 言語と、これの最適化である。

(2)　原子的トランザクション（同時実行制御と復旧制御）　複数利用者からのトランザクションに対して、データベースのインテグリティ制約を保つとともに、システムのス

ループットの向上がデータベース管理システム（DBMS）により行われている。

(3) 二次記憶へのデータの効率的格納と高速アクセス方法の提供。索引とハッシング等のアクセス方法、ポインタ、クラスタリング等の物理構成技法がこれである。

オブジェクト指向システムは、以前述べたように、オブジェクト、メッセージ・パッシング、階層構造、継承といった概念に基づいたシステムである。

オブジェクト指向データベースシステム(OODBS) とは、データベースシステムの(1)～(3)の性質と、オブジェクト指向概念が結合されたものである。

C．要求

現在のリレーショナル・データベースシステムは、CAD 等での要求に対して、十分な機能を提供できない。こうした分野では、以下の機能が要求されている。

(1) 複雑な情報のモデル化機能。 データの要素間での汎化（is-a）の関係、集積化（part_of)の関係が提供される必要がある。これらにより、種々の抽象化のレベルで、データ構造を表現できる。

(2) 意味の扱い。 複雑なデータを設計し、操作するためには、実世界での意味をデータベースに取り組んでいく必要がある。従来のリレーショナル・モデルでは、オブジェクト間の関連、制約を実現することは容易ではなかった。これに対して、オブジェクト指向モデルでは、オブジェクト階層、継承を用いることにより、オブジェクト間の制約を表現できる。

(3) 動的なスキーマの拡張性。 例えば、設計の変更等に対して、データベースを容易に変更し、拡張できる必要がある。従来のデータベースシステムでは、スキーマの変更は、容易でない。オブジェクト指向モデルでは、クラスの定義により、スキーマを容易に変更できる。

(4) 一致性の管理。 データ構造が複雑になるにつれて、データの一致性を保つことが重要になる。一致性の制約は、データ型、値の範囲、設計条件等により与えられる。こうした一致性の条件を、オブジェクト指向モデルでは、プログラムでチェックするのではなく、データ構造としてチェックできる。

(5) 大量データの管理。 設計等の新しい応用では、大量のデータが管理される必要が

ある。このためには、データを、複数のシステムに分散できる必要がある。オブジェクト指向モデルでは、オブジェクト単位に分散させることが容易であり、分散システムのフレームワークとなる。

(6) メタ・データの管理。 データについてのデータを管理できる必要がある。

(7) データの共有。

(8) バージョン管理。 設計では、特に、データの版の管理が必要である。

(9) 利用者間の通信。 例えば、設計者間での通信を提供できる必要がある。以下のような種々の通信のモードが必要である。

1）ロック通信。利用者に、オブジェクトのロック状態を通知する。

2）更新通信。オブジェクトの更新状況を通知する。

3）競合通信。トランザクション間での競合について通知する。

4）交渉通信。利用者間での合意等をとるための通信である。

(10) 柔軟なトランザクション管理。read/writeについての同期制御だけでなく、意味的な制御が必要である。

(11) 高速なデータ・アクセス。

(12) データベース・プログラム言語。 従来のリレーショナル・モデルでは、データ言語とプログラム言語とは異なっていた。これに対して、一つの言語により、統合的に、データ操作とプログラムが行える必要がある。

(13) 互換性、拡張性、統合化。 既存の種々の計算機資源を利用できる必要がある。

(14) グラフィクス・インタフェース データ構造のブラウジングや、更新を容易に行える必要がある。

D. 利点

オブジェクト指向データベースシステムは、リレーショナル・データベースシステムに対して、以下の利点を持っている。

(i) 実世界との対応がよくとれる。実世界のエンタティが、オブジェクト指向データベースシステムのオブジェクトと対応付けれる。また、リレーショナル・モデルでは、実世界のエンタティの『動作』をモデル化できないが、オブジェクト指向モデルでは、こ

れをメソッドとしてモデル化できる。

(2) オブジェクト指向モデルは、柔軟である。リレーショナル・モデルで必要となる正規化等の複雑な概念が不要である。

(3) スキーマの拡張が容易である。オブジェクト指向モデルでは、汎化と継承により、実世界の意味を取り込み、スキーマを構成したり、変更することが容易である。

(4) 統合言語が提供される。リレーショナル・モデルでは、SQL のようなデータ言語と、C等のプログラム言語が相違している。しかし、オブジェクト指向モデルでは、データ操作とプログラムを同じ言語で行える。

(5) オブジェクトの識別機能がある。リレーショナル・モデルでは、値が調べられるだけであるが、オブジェクト指向モデルでは、オブジェクトの同一性と、オブジェクト値の同値性が区別される。

(6) トランザクション実行の並行度を高められる。単なるreadとwrite 演算だけではなく、オブジェクトのメソッドのレベルでの同期制御を行えることから、トランザクション処理の並行度を高めれる。

(7) より強力な質問言語が提供される。

(8) 協調作業を支援できる。リレーショナル・モデルよりも、より複雑なデータ構造を提供できることから、よりデータの共有が容易となる。

E．システム構成

オブジェクト指向データベースシステムは、一般的にクライアント・サーバモデルの形態をとっている［図２－12］。複数のワークステーションが、ローカルエリアネットワーク(LAN) 等の通信ネットワークにより結合された分散環境で利用できる。サーバは、オブジェクトの定義、操作を行う基本演算を提供するとともに、複数の利用者に対する同時実行制御、障害に対する復旧を行う。クライアントは、利用者とサーバ間のインタフェースであり、応用の開発と、実行を行わせる。

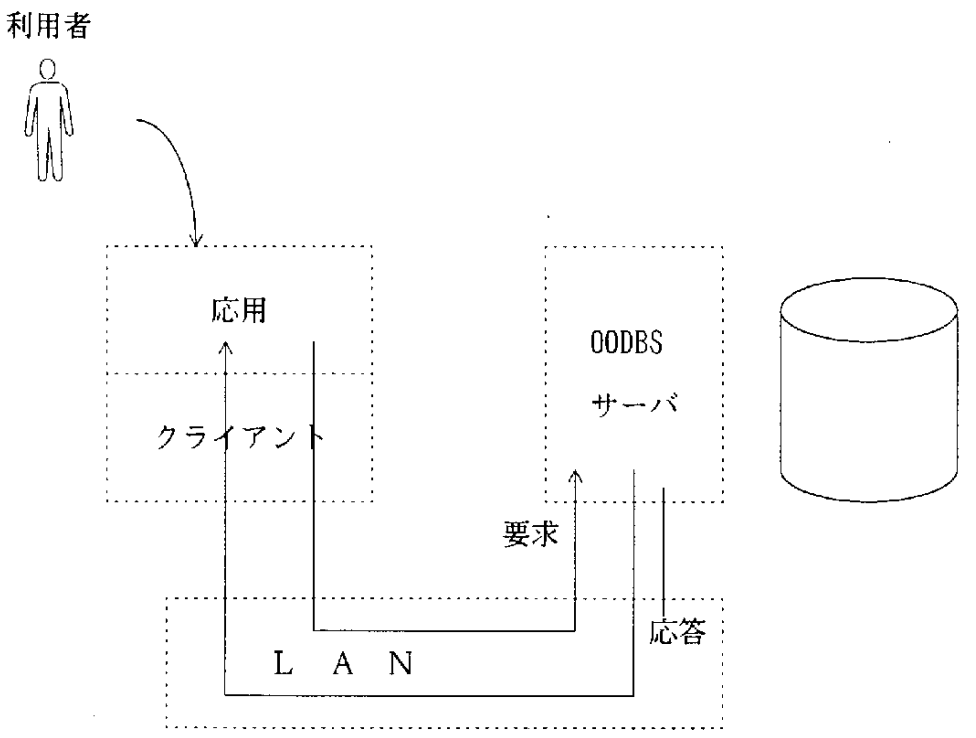


図2-12 サーバ・クライアント・モデル

システムとしては、以下の種類がある。

(1) 多クライアント・単一サーバ

(2) 多クライアント・多サーバ

今後のオブジェクト指向データベースシステムは、(2)のタイプになっていき、分散した複数のサーバ内のオブジェクトを、クライアントを通じてアクセスできる。

オブジェクト指向データベースシステムの実現方法としては、つぎの方法がある［図2－13］。

(1) リレーショナル・データベースシステム上にオブジェクト指向インタフェースを設ける。

(2) オブジェクト指向データベースシステムをOS上に直接実現する。

(1)では、複合オブジェクトを示す非正規形テーブルの実現が必要となる。データベースシステムとして、SQL といった標準モデルを提供する必要があることから、(1)の型の実現が有効である。オブジェクト間の汎化、集積化の関係を、データベースシステムとし

て記憶するために、複合オブジェクトが用いられる。

オブジェクト指向言語（C＋＋，Smalltalk）

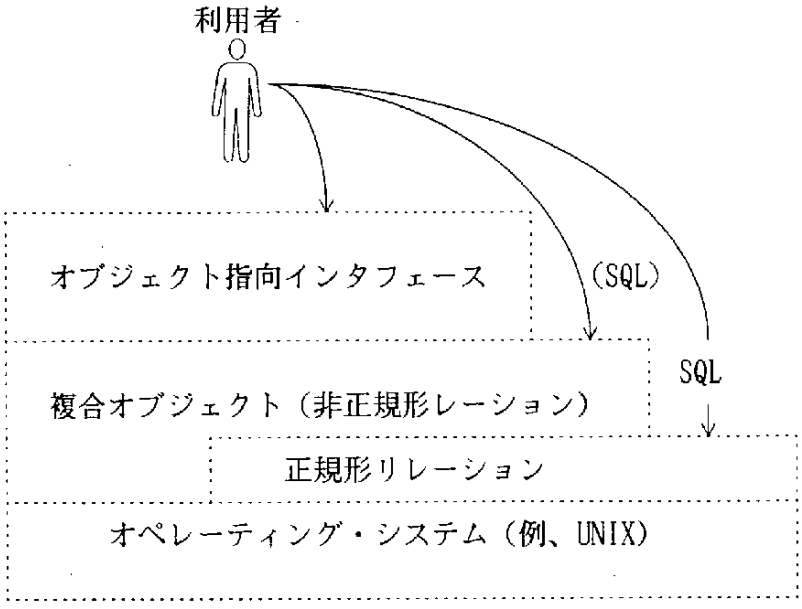


図2-13　オブジェクト指向データベースシステムの構成

F．言語

オブジェクト指向データベースシステムは、オブジェクト指向の言語がデータベースの定義、操作とプログラムを統合的に扱うものとして提供される。現在のシステムでは、C＋＋、Smalltalk といった既存のオブジェクト指向言語を拡張したものが用いられている。

〔例〕例えば、Gemstoneでは、Smalltalk を拡張した言語OPALが用いられている。例として、以下のスキーマのオブジェクト（複合オブジェクト）を考える。

```
Employee (
Empname (firstName/String, lastName/String)/PersonName,
ssNo/SmallInteger,
Address (stNumber/SmallInteger, street/String,
city/String)/HomeAddress,
Salary/SmallInteger )
```

これに対するメッセージ式は、以下のようにSmalltalk と類似している。基本形式は、<receiver><message >である。

1）単項メッセージ　　emp firstName

2）二項メッセージ　　8 * 3
(emp1 salary) ≦ (emp2 salary)

3）キーワード・メッセージ

```
anArray at:3  put: 'Ross'
```

メソッドも、以下のように、Smalltalk と同様な形式であらわされる。

```
increaseBudgetBy: aPercentage upIo: aLimit
    budget := budget + (budget * (aPercentage /100)).
    (budget > aLimit) ifTrue: [budget := aLimit].
    ^budget
```

最近のシステムでは、C++が言語として多く用いられている。
オブジェクト指向プログラム言語に対して、オブジェクト指向データベースシステムでは、オブジェクトを恒久的（persistent）にする機能が必要である。

G. 多重継承

多重継承では、継承する名前（変数名、メソッド名）競合の解決が問題となる。ORION/ITACSAでは、以下のような競合解決方法がとられている。

(1) 嗜好継承（preferential－inheritance）：利用者が、各オブジェクトに対して、性質を継承する上位クラスを選択する。
(2) 継承語の改名（rename－after －inheritance）：継承された変数名が競合するとき、導出クラス内で改名する。

Gemstoneでは、多重継承は、提供されてない。

F．集積オブジェクト

オブジェクト間での集積関係part-of(または、複合関係）により、オブジェクトの階層を構成する。ORION/ITACSAでは、オブジェクト間のpart-of の関係に対して、種々の意味が与えられている。

(1) 値伝播：集積オブジェクトの値が、要素オブジェクトに継承する。例えば、車を示すCar オブジェクトの要素であるDoorを考える。Doorは、Car の変数color が赤であるならば、この値をとる。

(2) クラスタリング：要素オブジェクトが、二次記憶内に近接して記憶できる。クラスタリングの単位となっている。

(3) スキーマの再定義：オブジェクトを、集積型とそうでないものとの間で型を変更できる。

(4) ロック：表に示すようなロックのモードを提供している［図２－14］。

IS： クラス定義をIntension 共有モードでreadに対して、ロックする。
Sモードで、例（インスタンス）オブジェクトをロックする。

IX： クラス定義を、Intension 排他モードで、更新に対して、ロックする。例オブジェクトをXモードでロックする。

X： クラス定義を更新に対して、ロックし、その例を暗黙に、Xモードでロックする。

S： クラス定義をreadに対してロックし、その例を暗黙に、Sモードでロックする。

SIX ：クラス定義をreadに対してロックする。そのクラスの例を暗黙に、Sモードでロックする。例を用に、Xモードでロックする。

	IS	IX	S	SIX	X
IS	○	○	○	○	X
IX	○	○	X	X	X
S	○	X	○	X	X
SIX	○	X	X	X	X
X	X	X	X	X	X

○：両立
X：非両立

図2-14 ロックの互換関係

H. 動的なスキーマ拡張

クラス定義等の変更を、応用を変更または停止することなしに、行える必要がある。スキーマ拡張には、以下のような変更がある。

(1) オブジェクトの中身の変更。 変数とメソッドの追加、削除、改名、定義域の変更、デフォルト値の変更等がある。

(2) クラス間の関係の変更。

(3) クラス束内の構成の変更。新しいクランイアントの追加、クラスの除去、クラス名の変更がある。

I. 記憶空間管理

二次記憶の管理方法について考える。ディスク内のページの確保、解放、ページのディスクとクランイアント間での転送、オブジェクトのクラスタリング、オブジェクト集合の索引、メモリ内でのオブジェクトのバッファリング等を考える必要がある。ORION/ITACSAについて考える。

(1) 各オブジェクトは、<class －id, instance－id>を識別子 (UID)としている。オブジェクトは、クランイアント内のメモリ内のバッファにディスクから転送されて利用される。

(2) クラスの全例と、オブジェクト集合を、ディスク内で近接に配置することにより、クラスタリングできる。

(3) クラス階層索引というB$^+$ 木と類似した索引を提供している。利用者が指定したクラスを根とした階層内の全クラスに共通の属性に対して定義される索引である。

(4) 共用と仕様という二種のデータベースを提供している。共用データベースは、全利用者で利用可能である。一方、私用データベースは、許可された利用者に対してのみ利用できる。データは、データベースから、取り出したり (check in) 、データベースに転送 (check out)できる。複合オブジェクトがcheck in/outの単位である。版を付けれるオブジェクトは、私用データベースにcheck out されるときに、版がつけられる。

J．変更通知

クランイアント間の通信は、分散環境では重要である。クランイアントに対して、データベースの変更を通知する機能が必要となる。通知方法としては、以下のものがある。

(1) メッセージ：変更通知を、関連する全クランイアントに通知する。

(2) フラグ：利用者がオブジェクトアクセスを行う時に、このオブジェクトの変更があったかどうかのチェックを行う。

(3) 即時通知：変更のたびに、変更通知をクランイアントに直ちに行う。

(4) 遅延通知：クランイアントが、他のクランイアントに一定時間後に変更通知を行う。

K．版管理

ORION/ITACSAでは、以下の種類の版を提供している。

(1) 中間版（transient）：作業中の版である。今後、かなり変更されるものである。

(2) 作業版（working）：公開するまえのもので、中間版よりも固定している。

(3) 公開版（released）：他の利用者により共用できる版である。

L．システム例

オブジェクト指向データベースシステムとしては、これまでにいくつかのシステムが実現され、利用されてきている。これらについて、簡単に述べる。

(1) ORION/ITACSA

ORION は、MCC(Microelectronics and Computer Technology Corporation) における ACT(Advanced Computer Technology) プログラムにより開発された次世代のデータベース管理システムである。1985年に、このプロジェクトは開始された。ORION は、CAD 等の設計応用のために、オブジェクト指向言語とデータベースシステムとの競合を行うことが目的であった。ORION は、次のように開発されてきている。

ORION-1　：単一利用者用のシステムである。（1987）

ORION-1SX ：多利用者のシステムである。サーバ・クライアント構成をとり、多クライアントで、単一サーバシステムである。（1988）

ORION-2　：多利用者のシステムで、多サーバで、多クライアントの分散型システムである。（1989）

ORION の商用版は、ITACSAであり、Itacsa社から販売されている。

ORION は、Common lisp により実現されている。ORION では、Common lisp により応用を容易に実現できる。ORION は、Symbolics 社の3600 LISP マシンでの動作に適しているが、UNIXワークステーションでも動作する。ITACSAは、Common lisp に加えて、Cに対するインタフェースも提供している。さらに、FORTRAN のコードを、ITACSAのメソッドとして利用できる。ITACSAは、Sun 等のUNIXワークステーションで動作する。

(2) ObServer/ENCORE

ObServerは、Brown 大学で開発された汎用のオブジェクト・サーバ・システムである。これは、二次記憶の管理、クラスタリング、オブジェクトのロック、トランザクションの管理機能を持っている。ENCODE(Extensible and Natural Common Object Resource)は、ObServerのフロントエンドであり、オブジェクトの定義と操作を行える。

ObServerとENCOREは、ともにCで開発され、UNIXのもとで動作する。ENCOREは、Cへのインタフェースを提供している。Cにより、応用を作成できる。

(3) GEMSTONE

GEMSTONEは、Servio社で開発された商用のオブジェクト指向データベース管理システムである。商用システムとしては、もっとも歴史の古いものの一つである。Smalltalk を拡張したオブジェクト指向言語により、オブジェクトの定義、操作を行う。

GEMSTONEは、Cにより実現されていて、Sun 等のUNIXワークステーションで動作する。

GEMSTONEの言語であるOPALは、Smalltalk に基づいている。C言語のインタフェースも提供しているが、十分ではない。

(4) ONTOS

ONTOS は、Ontologic 社により開発された商用システムであり、VBase とはじめは呼ばれたシステムである。オブジェクト指向言語であるC＋＋を拡張した言語を提供している。

ONTOS は、C＋＋により、実現され、UNIXワークステーションで動作する。

(5) ObjectStore

ObjectStore は、Object design 社により開発された商用システムである。C＋＋を拡張した言語を、提供している。

ObjectStore は、CとC＋＋により実現されている。ObjectStore では、いくつかの点で、C＋＋をう拡張している。例えば、new 演算子の恒久版を持ち、オブジェクトをヒープ領域とともに、二次記憶に生成できる。

(6) VERSANT

VERSANT は、Versant Object Technology 社により開発された商用のシステムである。これも、C＋＋を拡張した言語を利用者に提供している。

VERSANT は、C／C＋＋により実現され、Sun 等のUNIXワークステーションで動作する。C＋＋とCにより応用プログラムを作成できる。

2.2 グループウェアにおけるデータベースシステムの自律性と協調

2.2.1 グループウェアにおける分散型データベースシステム

情報システムでは、複数のデータベースシステムが通信網で結合された形態となる。複数のデータベースシステムから構成されるシステムを、一般に分散型データベースシステムと呼ぶ。特に、自律的なデータベースシステムを連邦的に統合する問題について考える。

A. はじめに

情報システムでは、複数のデータベースシステム（DBS)が通信網で結合された形態となる。複数のデータベースシステムから構成されるシステムを、一般に分散型データベースシステム（DDBS）と呼んでいる。ここでは、分散型データベースシステムとは何かとその動向について考える。通信網では、OSI 等を中心として、広域網、ローカルエリア網で国際標準化が進み、どのメーカの製品も相互結合できるようになってきている。さ

らに、FDDI、ATM/B-ISDN、HIPPI 等のGbpsレベルの超高速網の研究と開発が行われている。

データベースシステムでは、CODASYL により行われてきた網型モデルの国際標準化に続いて、リレーショナルデータベースシステムのSQL の国際標準化が行われ、SQL を提供する種々のリレーショナルデータベースシステムが広く用いられている。又、従来のデータベースシステムが事務処理用の簡単なデータ構造と操作を提供してきたのに対して、最近ではCAD 等の新しい応用でより複雑なデータ構造と操作が求められている。このために、オブジェクト指向モデル [KIM89] 、ERモデル [CHEN86] のような意味的なモデルが示されている。さらに、ワークステーション用のデータベースシステムでは、クランイアントとサーバにユーザインタフェース機能とデータベースの基本操作機能を分離した構成が取られ、複数のサーバを用いた応用が作られてきている。こうしたことから、通信網により相互結合された複数のデータベースシステムを統合的に利用できることが求められてきている。

B．定義

分散型データベースシステム（DDBS）とは、通信網により相互結合された一つ以上のデータベースシステム（DBS)から構成されるシステムである。分散型データベースシステムの各要素を節（node）システムとする。

データベースシステムとは、ある型のデータモデルを提供するシステムである。データモデルとは、データ構造、演算、インテグリティ制約で与えられる。リレーショナルデータベースシステムでは、テーブルと国際標準データ言語のSQL [SQL] を提供しているシステムである。この他に、従来から利用されているネットワーク型のデータベースシステムがある。最近では、オブジェクト指向モデル [KIM89] 、ERモデル [CHEN86] の意味的モデルもある。

データベースシステムは、データベース（DB）とデータベース管理システム（DBMS）から構成される。データベースは、データの集合である。DBMSは利用者とデータベース間のインタフェースであり、利用者からの操作要求を受け付けてデータベースの操作を行う。DBMSを論理的にみると、利用者にあるデータモデルを提供するシステムである。DBMSは、システム的には次の機能を持つ。

(1) 複数利用者に対して、データベースのインテグリティ制約を保つと同時に、スループットを向上にさせるための同時実行制御。

(2) 障害に対してデータベースのインテグリティ制約を保つためのフォールトトレランス制御。

(3) 不正利用者によるデータベースシステムの不正利用を防止する安全性制御。

(4) 利用者の質問をなるべく応答時間が短くなるようにする最適化。

(5) ユーザインタフェース。

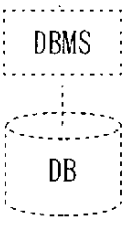
データベースシステムの形態について考えると、以下の種類がある［図２－15］。

１）集中型データベースシステム

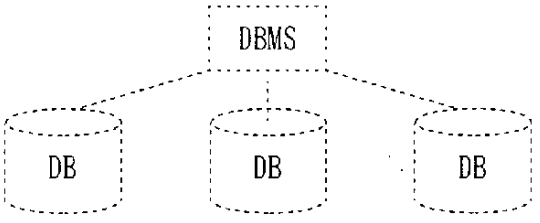
２）分散型DBMS

３）多データベースシステム

(1) 集中型DBMS



(2) 分散型DBMS



(3) 多データベースシステム

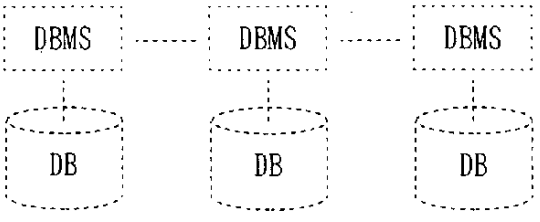


図2-15 分散型データベースシステム

集中型データベースシステムでは、データベースとDBMSが同一の計算機内にある従来型のデータベースシステムである。

分散型DBMSでは、データベースは複数の計算機に分散しているが、DBMSは一つである。各データベースは、その計算機内に管理機能を持つが、分散型データベースシステムとして一つのDBMSが存在する。システム全体の性能、信頼性、可用性等［CERI84］を向上させるためにデータを複数の節システムに分散する。

多データベースシステムは、データベースが複数の節システムに分散し、各データベースはそれ自身のDBMSを持つ場合である。各要素がDBMSを持つことは、各々が何らかの自律性を持つことである。各工場、部門等毎に自然発生的にデータベースシステムが構成されてきた場合や、通信網を通して種々の情報サービスを利用する場合がこれにあたる。

分散型データベースシステムは、(2)と(3)のシステムを指す。また、狭義には、多データベースシステムを指す。本調査研究では、今後の分散型システムの中で重要である(3)について考える。

2.2.2　多データベースシステムの異種性・自律性・分散性

多データベースシステムを考えるとき、各節システムについて、(1)異種性（heterogeneity）、(2)自律性（autonomy）、(3)分散性（distribution）が重要である。

A. 異種性

多データベースシステムは、複数のデータベースシステムから構成される分散型データベースシステムである。ここで、節システムの異種性が重要であり、異種性については、データベースとDBMSとの二点について考える必要がある。

データベースの異種性には、構文的異種性と意味的異種性とがある。構文的異種性とは、同一のデータベースを表現する方法の差で、以下の点がある。

(1)　データ構造　　例えば、同一データをリレーショナル型モデルでは関係として表現し、ネットワーク型モデルではレコード型として表現する場合の表現の相違である。また、外来キーと親子関係も例である。

(2)　データ言語　　二つのデータベースシステムのデータ構造が同一でも、その言語が異なる場合である。例えば、あるリレーショナルデータベースシステムが、一つはSQL

を、他方はQUELを用いている場合である。

(3) データ型　同一の属性（項目）を異なったデータ型で表現している場合である。例えば、社員番号を、あるデータベースシステムでは整数で、他方では文字列で示している場合である。

(4) データ単位　同一の値を異なった単位（例えば、グラムとポンド、キロメートルとマイル）で示している場合である。

(5) データ構造の基本要素　同一の情報を異なったデータモデル要素で示している場合である。例えば、名前を、あるデータベースシステムでは苗字とあわせて一つの属性により表現しているが、他のデータベースシステムでは二つの属性により表現している場合である。

二つのデータベースシステムについては、以上の点について何らかの相違があるとき、互いに構文的に異種であるとする。

意味的異種性は、次の場合に生じる。

(1) 同一のデータについて、意味、解釈、利用方法が相違する。

(2) 構文的に同一のデータが複数の意味を持つ。

二つのデータベースシステムが、(1)又は(2)の相違を持つとき、互いに意味的に異種であるという。例を考える。

［例］二つのリレーショナル型データベースシステムＡとＢがあり、各々本と車の情報についてのデータを持っているとする。Ａ内のテーブルBookの属性costは各本の価格を示し、Ｂ内のテーブルCar の属性costは車の価格で消費税が含まれた額を示しているとする。二つの属性は構文的には同種であるが、意味的には異なっている。これは、二つの属性の意味の相違である。

複数のデータベースシステムの間に意味的な異種性が存在することを見つけることは容易ではない。特に、異なったデータモデルのデータベースシステム間では、特に困難である。意味的異種性についてはまだ充分な研究がなされていず、今後の課題である。

次に、DBMSの異種性について考える。DBMSの異種性には、論理的なものと、システム的なものとがある。論理的異種性とは、データモデル、即ちデータ構造とデータ言語の相違である。例えば、関係型モデルを提供するDBMSと、CODASYL モデルを提供するDBMSの

間の相違である。システム的異種性とは、DBMSの同時実行制御、復旧等の信頼性制御、安全性の制御、最適化方法等のDBMSの機能の相違である。

B．自律性（autonomy）

次に大切な点は、各データベースシステムが、分散型データベースシステム全体との間で、どの程度の自律性を持つかである。データベースシステムの自律性を考えるとき、設計、関連、通信、実行の４点［VEIJ88］が大切である。設計について自律的であるとは、データベースシステムの設計を他のデータベースシステムと独立に行える事である。具体的には、以下について独立に決定を行える事である。

１）定義域、即ち、データベースシステム化される情報。

２）データ要素の表現（データ構造、データ言語）と名前付け。

３）データの意味的解釈または概念化。

４）データを管理するための制約（インテグリティ制約、直列可能性等）。

５）システムの機能。

６）他のシステム間での関連付けと共有。

７）実現方法（例、ファイル編成、同時実行制御方法）。

関連（association)について自律的であるとは、データベースシステム内のデータを他のデータベースシステム内のデータと関連を付ける又は関連付けを解消することについて各データベースシステムが決定できることである。

通信について自律的であるとは、他のデータベースシステムと通信するかしないかを自分で決定できることである。通信の自律性を持つシステムは、他のシステムからの通信に対して、何時どのようにして応答を返すか自分で決めることができる。

実行について自律的であるとは、他のデータベースシステムからの演算を、実行するかしないか、どのような順序で実行するかを決めれることである。

C．分散性（distribution）

データは、複数のデータベースシステムに分散される。分散型DBMSで、データが複数の要素に分散されることの利点は、信頼性、可用性、効率、安全性等の点である。一方、多データベースシステムでは、既にデータベースが各要素で構成されている。

2.2.3　自律型分散データベースシステムの統合化

分散した複数のデータベースシステムを利用者が統合的に利用するためは、分散性、異種性、自律性と独立に利用できる必要がある。

A．連邦化

複数のデータベースシステムの統合化には、次の間に種々のレベルがある。

(1)　完全統合化　各データベースシステムをそのローカル利用者として操作できず、ただ一つの全体的なデータモデルを通してのみ利用できる。

(2)　非統合化　利用者は、各データベースシステムをローカルに利用できる。

完全統合化では、分散型データベースシステムとして一つの管理者が存在し、これが全てを管理する。非統合化では、全体の管理者が存在しない。

(1)と(2)の間にあるレベルを連邦化（federation）［HEIM85］という。ここでは、連邦化について考える。連邦化により、複数のデータベースシステムから連邦スキーマが構成される。

多データベースシステムでは、一つ以上の連邦スキーマが存在できる。連邦スキーマが一つしかないシステムを単一連邦スキーマのシステムとし、複数あるものを多連邦スキーマのシステムとする。粗結合システムは多連邦スキーマのシステムである。密結合システムは、単一の場合も多の場合もある。

連邦スキーマがどのような頻度で生成されるかについて、以下の二つがある。

(1)　動的連邦化。

(2)　静的連邦化。

動的連邦化では、頻繁に連邦スキーマが定義され、除去される。又、連邦スキーマを定義し除去することは自由に行われる。

一方、静的連邦化では、連邦スキーマは、一度定義されると基本的に変更されない。静的連邦化では、全体としての管理のもとで、連邦スキーマが定義され、保守される。

B．自律性のレベル

まず、連邦化を誰が管理していくかにより、(1)粗結合と(2)密結合の二種がある。粗結合では、各利用者が連邦化を行う。利用者が必要とするデータを自分の目的に合うように

複数のデータベースシステムから統合する。粗結合の分散型データベースシステムを、相互運用可能（interoperable)又は多データベースシステムという場合もある［LITW82］。例として、MRDSM ［LITW85］、CALIDA［JACO88］、OMNIBASE［RUSI89］等がある。

密結合システムでは、一人の連邦化管理者（DBA)により、連邦化、保守管理が行われる。例として、SIRIUS-DELITA ［LITW82］、PRES I* ［DEEN］、DDTS［DWYE87］、MERMAID ［TEMP87］、MULTIBASE ［LAND82］、JDDBS ［TAKI83］等がある。

連邦化では、複数のデータベースシステムから必要なものを選択し、統合化される。この結果、連邦スキーマが構成される。粗結合システムでは、スキーマ輸入（sehemaimportation）［HEIM85］、視野定義［MOTR81］、多データベース質問［LITW87］といわれる手続きにより連邦スキーマが構成される。この手続きは、密結合システムでは、スキーマ統合［BATI86］といわれる。

粗結合システムは、検索中心のシステムに適している。各利用者毎に視野（連邦スキーマ）が定義されるので、これを通して更新することは、視野更新と同様の問題が生じるので、困難である。これに対して、密結合システムでは、システム全体として連邦化されるので、更新に対してインテグリティ制約を保つことが容易である。

2.2.4　分散型データベースシステムの参照モデル

A．スキーマ階層

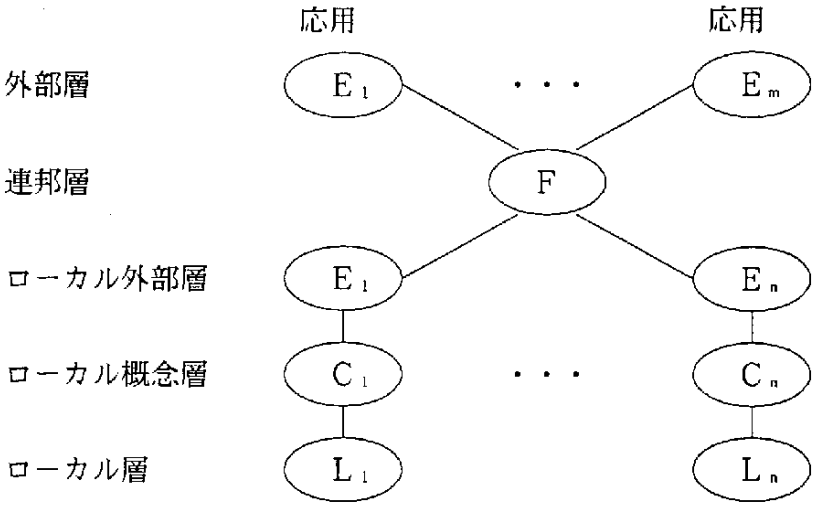


図2-16　五階層構造（単一連邦スキーマ）

データベースシステムでは、標準的な参照モデルとして、データの物理的と論理的独立性を達成するために三階層のモデル［TSIC78］がANSIにより示されている。これに対して、分散型データベースシステムでは、分散性、異種性、自律性からの独立性を考える必要がある。これまでに、四層［DEEN85, TAKI83, TEMP87］、五層［DEVO82, LITW82］等の階層モデルが提案されている。四階層モデルでは、分散性と異種性との独立性を達成することが目的である。自律性まで考えると以下に述べる五階層の構造が必要になる［図２－16］。

［スキーマ階層］

(1) ローカル層

ローカル層は、既存のデータベースシステムの概念層である。一般に、ローカル層のスキーマは、各データベースシステム固有のデータモデルにより示されていて、異種である。

(2) ローカル概念層

ローカル概念層は、分散型データベースシステム全体で共通のデータモデルにより、ローカル層スキーマを表現したものである。このとき、ローカル層のスキーマに表現されていないデータベースの持つ意味が付加される。

(3) ローカル外部層

ローカル外部層は、他のデータベースシステムにより利用できるデータを示したものである。従って、ローカル概念層の一部であり、ローカルなデータベースに対するアクセス制御情報も持つ。データベースシステムの自律性を管理、制御することを容易にするためのものである。これは、輸出（export）層ともいわれる。

(4) 連邦層

連邦層は、複数のローカル外部層を統合したものである。連邦層のスキーマは、データがどのデータベースシステムにあるかを示す情報（分散スキーマという場合もある）も持つ。連邦スキーマは、輸入（export）スキーマ［HEIM85］、全体（global）スキーマ［TAKI83, LAND82］、全体概念（global conceptual)スキーマ［LITW82］と呼ばれるものである。

(5) 外部層

外部層は、各応用毎に定義される。応用に対して、必要とするデータを示すデータモデルを提供し、種々の意味と付加的な制約を与え、アクセス制御を行うことが目的である。

粗結合のシステムでは、図２－17に示すように複数の連邦スキーマを持つ構造となる。

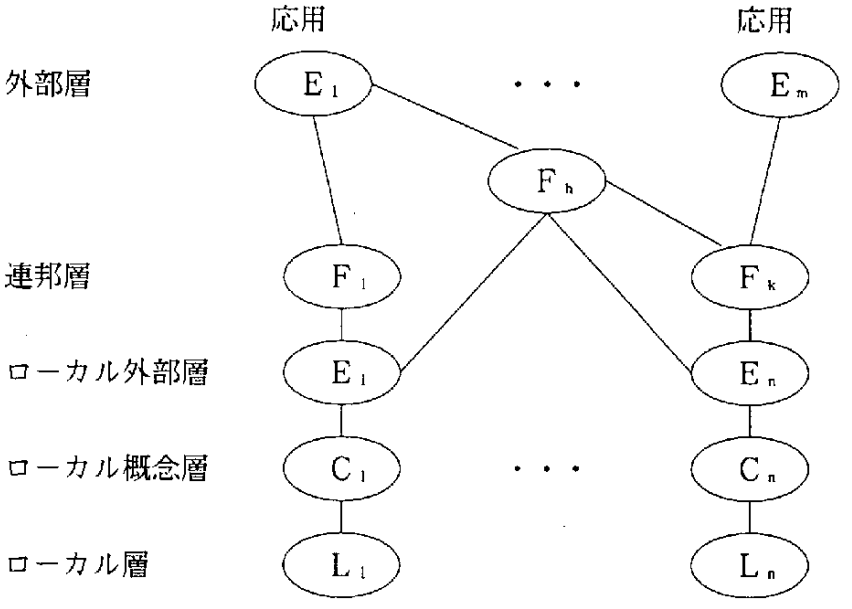


図2-17　五階層構造（多連邦スキーマ）

これらの５層の他に、以下の目的のために予備的なスキーマを持つ場合がある。

１）既存のデータベースシステム内に存在しないけれども、連邦化するために必要となるデータ。例えば、ローカル・スキーマに示されていない意味的情報。

２）種々の非互換性（incompatibility)を解決するために必要なデータ。例えば、単位の変換のためのデータである。

３）質問処理と最適化のための統計情報。

Multibase ［SMIT81］では、予備スキーマ内に１）と２）の情報を保有している。Mermaid［TEMP87］は３）の情報を予備スキーマとして持っている。

ローカル層と外部層で種々の異種のデータモデルが提供される。しかし、ローカル概念、ローカル外部層、連邦層では、同一の共通モデルが用いられる。共通モデルとしては、高水準の意味を扱えるモデルが必要となる。現状では、ERモデル［CHEN76, ELMA85］が主に用いられている。

分散型データベースシステムの管理者には、ローカルDBA と連邦DBA の二種がある。前者は、各データベースシステムを管理し、後者は分散型データベースシステム全体を管理する。

2.2.5　システム構成とトランザクション管理

次に分散型データベースシステムのアーキテクチャについて考える。

分散型データベースシステムは、５つの階層間の写像、外部野 ⇔ 連邦、連邦 ⇔ ローカル外部、ローカル外部 ⇔ ローカル概念、ローカル概念 ⇔ ローカル写像を行う論理的なプロセッサから構成される。各プロセッサは、隣接する階層間の写像を行うものであり、以下の二つから構成される。

(1)　演算の変換を行う、演算変換器。

(2)　データ構造の変換を行うデータ構造変換器。

また、変換を行うために必要となる情報を保持するための写像情報がある。図２－18は、上位層Ａと下位層Ｂ間の写像を示している。

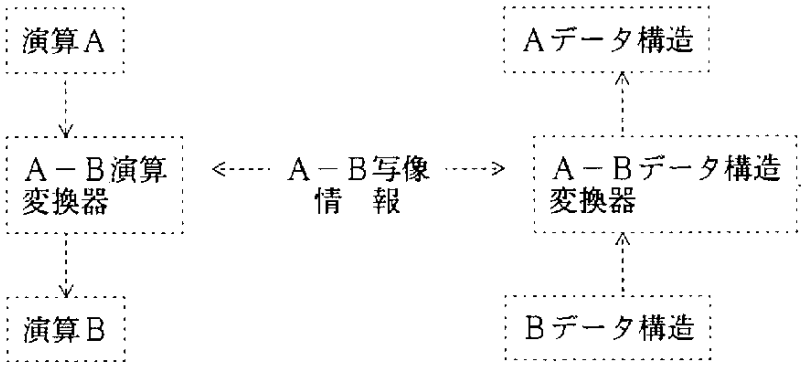


図2-18　層間写像

(1)　ローカル概念（le）変換

LCは、ローカル層とローカル概念層との写像を行う変換器である。LCP の目的は、異種

のデータベースシステムに対して、共通のデータモデルを提供することである。即ち、利用者に対して、各データベースシステムのデータ言語とデータ構造の相違を意識させないことである。

演算変換は、共通モデルの言語を各データベースシステム固有の言語に変換する。例えば、SQL をCODASYL コマンドへの変換［ZANI79, ONUE83］、QUELをCOBOL DML プログラムへの変換［TAKI80］がある。又、ローカル層に対する演算により得られた結果を共通モデルのデータ構造に変換する必要がある。例えば、CODASYL データベースシステムから得られたデータを関係に変換することである。

(2) ローカル概念－ローカル外部（LE）変換

LE変換器は、ローカル概念層とローカル外部層間の写像を行う変換器である。ローカル外部層は、ローカル概念層と同一のモデルを用いており、その一部である。

(3) ローカル外部－連邦（EF）変換

EF変換器は、各ローカル外部層と連邦層間の変換器である。この目的は、利用者に、分散性からの独立性を提供することである。即ち、利用者は、複数のデータベースシステムをあたかも一つのデータベースシステムのように見せることである。ローカル外部層と連邦層のデータモデルは同一である。連邦層の演算を、複数のローカル外部層の演算に変換（分割）し、各データベースシステムから得られたデータを結合することが必要である。

1）質問分割と最適化　連邦層の質問を、各データベースシステムのローカル外部層上の質問に分割すると同時に、その実行を最適化することである。

2）データ結合　複数のデータベースシステムから得られたデータを結合する。

3）トランザクション管理　複数のデータベースシステムに対するトランザクションを実行するための同時実行制御、コミットメント制御である。

(4) 連邦－外部層（EE）変換

外部層の演算を連邦層の演算に変換し、その結果を外部層のデータ構造として導き出すことがEE変換器の仕事である。EE変換器は、外部層固有のデータモデルと共通のデータモデル間の変換である。基本的には、LC変換器と同じ機能を持つ。

粗結合システムでは、利用者と応用毎に複数の連邦スキーマが、動的に構成される。密

結合システムには、単一連邦スキーマと多連邦スキーマのシステムがある。図２－19に、多連邦スキーマのシステムのアーキテクチァを示す。

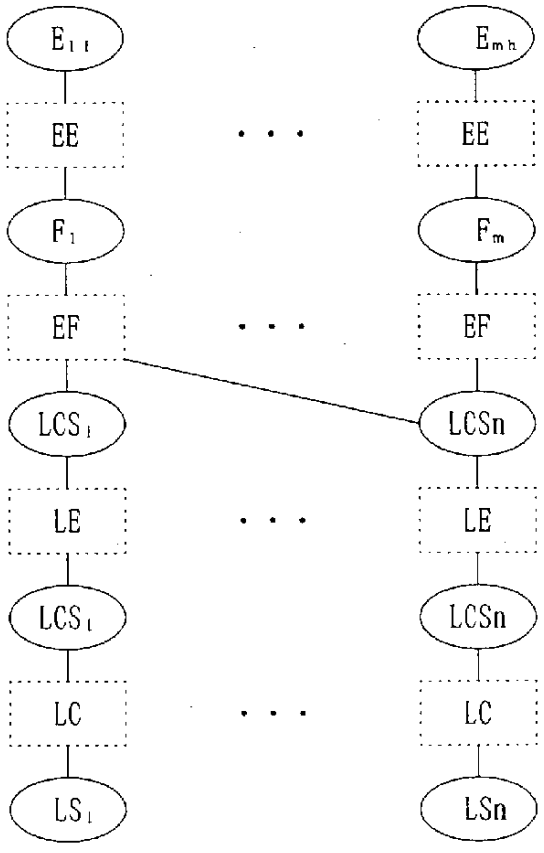


図2-19　アーキテクチァ

C．質問処理

完全統合化又は密結合システムでは、利用者は、データの所在場所、冗長性等の分散性を意識せずに、操作演算を質問言語等を用いて記述できる。しかし、粗結合システムでは、利用者は、複数のデータベースシステムに対して、質問を行わねばならない。この例として、MDSL［LITW87］、MSQL［LITW87］、GSQL［JAC085］がある。

分散型DBMSに対する質問の最適化については、［YU84］に述べられている。一般に、分散型DBMSと密結合システムでは、こうした最適化が行われるが、以下の理由で多データベースシステムでは最適化が困難である。

(1)　演算の実行コストは、各節システムにより異なっている。データベースシステムの自律性の度合いにより、ある節システムの演算の処理コストを知ることができない。ま

た、演算の実行コストは時間とともに、計算機の負荷等により変化する。

(2) 各節システムが提供する演算は異なっている。又、各節システムは、異なった最適化機構を持つ。

D. トランザクション管理

全体トランザクションとは、複数のスキーマ内のデータを原子的に更新する演算の集合である。ローカル・トランザクションとは、ある節システム内のデータを更新するトランザクションである。

複数のトランザクションの同時実行の正しさとしては、直列可能性［BERN87］がある。しかし、自律的なデータベースシステムでは、実行順序を全体の順序通り行えるとは限らない。このために、直列可能性を、弱めた擬直列可能性が示されている［DU89］。又、意味的な同時実行制御については、［GARC87］がある。

2.3 グループウェアにおける分散型データベースシステムの設計

2.3.1 設計機能とスキーマ変換

A. 設計機能

分散型データベースシステム内のスキーマ（例、ローカル外部、連邦スキーマ）を定義するためには、以下を行う必要がある。

(1) スキーマ定義　利用者の要求に合うように、スキーマを新たに定義するか、又は既存のスキーマ上にその副スキーマ（視野）を定義する。

(2) スキーマ変換　あるデータモデルのスキーマＳを、他のモデルのスキーマＴに変換する。このとき、ＴはＳと同値である必要がある。

(3) 交渉（negotiation）　ローカルDBA と連邦DBA の間で、データベースシステムの利用権についての交渉を行う。

(4) アクセス制御　交渉により定まった利用方法に従って、データベースシステムが利用されることを保障するための制御である。

(5) スキーマ統合　複数のスキーマから利用者の目的に合うように一つのスキーマを定

義する。

分散型データベースシステムの設計方法には、ボトムアップとトップダウンの二種がある。前者は、以下の手続きから構成される。(1)スキーマの変換、(2)ローカル外部スキーマの定義、(3)スキーマの統合、(4)外部スキーマの定義。

一方、トップダウン型設計は以下の手続きから構成される。(1)外部スキーマの定義、(2)スキーマの解析、(3)スキーマのを統合。

B. スキーマ変換

スキーマ変換とは、あるデータモデルM_1のスキーマS_1を、他のデータモデルM_2の目標スキーマS_2に変換することである。ここで、S_1とS_2は同値であることが要求される。分散型データベースシステムでは、ローカル・スキーマをローカル概念スキーマに、連邦スキーマを外部スキーマに変換する場合がスキーマ変換の例である。スキーマ変換には、以下の方法がある。

(1) 双方向スキーマ変換　　S_1からS_2への変換と同時に、逆の変換も用意する。

(2) 一方向スキーマ変換　　逆変換が出来ない場合である。この例として、CODASYL スキーマのリレーショナルスキーマへの変換がある［TAKI80, ZANI79］。ERモデルとリレーショナルモデル間の変換については、［ELMA86］が議論している。

(3) 変換をそのままでは行えない場合である。この場合には、DBS は、S_1を制限するか、他のスキーマから目標スキーマS_2内を合成する。

スキーマ変換では、スキーマは構文的に変換されるので、スキーマの意味は考慮されない。スキーマ統合、演算変換で必要となるローカル・スキーマの意味をスキーマ変換に取り組む必要がある。従って、共通モデルは、ローカル・スキーマ以外の付加的な意味を表現できる必要がある。例えば、汎化、集積化を表現でき、リレーショナルモデルの外来キーを表現できる必要がある。

2.3.2 グループ通信方式

A. 交渉（negotiation）

連邦DBA とローカルDBA は、ローカル外部スキーマに含まれるべきデータと、これに対

して行える演算について合意を取らねばならない。このための両DBA の対話を交渉とする。交渉の方法［HEIM85］には、以下のようなものがある。

［交渉プロトコル］

(1) ローカルDBA が連邦DBA に、ローカル概念スキーマを見ることを許す。しかし、それ以外の権限をあたえない。連邦DBA がローカル概念スキーマ内のデータをアクセスしたいときは、ローカルDBA に、メッセージを送信して依頼する。ローカルDBA はこれを受理するかどうかを決定する。

(2) ローカルDBA は、前もって、ローカル外部スキーマに対するアクセス権を定め、これにより連邦DBA に利用させる。

B．スキーマ統合

スキーマ統合とは、各データベースシステムのローカル外部スキーマから、連邦スキーマを定義することである。スキーマ統合はボトムアップ的設計であり、視野統合はトップダウン的設計である。［BATI86］によると、スキーマ統合は、以下の手続きから構成される。

(1) 予備統合（preintegration）

(2) スキーマ解析

(3) 適合（conformation）

(4) 再構成

スキーマ解析では、以下が行われる。

1）名前競合（例、同義語、同名異義語）、定義域競合（例、型）、構造的な差異、制約の相違、欠損データを明確にするために各スキーマを比較検討する。

2）スキーマ間の相互関係を明らかにする。

複数のスキーマを比較し、解析するためには、これらが共通のモデルで表現されている必要がある。そうでないと、比較解析の作業は困難である。この点からも共通モデルは、意味を充分に表現できる必要がある。特に、汎化の概念が重要である［DAYA84］。各ス

キーマ内では、類似又は関連したオブジェクトが種々の抽象化レベルで表現されているからである。

[例] スキーマAで、学生名を示す属性sname があり、スキーマBに社員名を示す属性ename がある場合を考える。これらは、人名nameを特殊化したものと考えることができる。従って、sname とename を汎化したnameを設けることにより、二つの属性を関連付けれる。

ERモデルを共通モデルとして考える。aとbを二つの属性とする。このとき、aとbの間には以下の関係がある [SHET89] 。

(1) aはbと同値（equivalent）である。

(2) aはbを含む（includes）。

(3) aはbと素である（disjoint）。

二つの（実体又は関連）型EとFの間には以下の関係がある [NAVA86, ELMA86] 。

(1) EとFは同値である。

(2) EとFは共通部分を持つ。

(3) EとFは素であるが、統合できる。

(4) EとFは素であるが、統合できない。

一般に、次の理由からスキーマ統合を自動化できない。

(1) 現在の意味データモデルでは、実世界の意味を完全に表現できない。

(2) スキーマとして表現されている以外の意味を取り組む必要がある。

(3) 実世界について複数の視野と解釈があり、これらは時間とともに変化する。

2.3.3 分散型データベース

A. DDTS

DDTS(Distributed Database Testbed System) [DWYE87] は、ハネウェルにより開発されたシステムである。DDTSは、5層の階層を持ち、共通モデルとしては拡張ERモデル [ELMA85] が用いられている。要素のデータベースシステムとしては、リレーショナルとネットワーク型のデータベースシステムがある異種の分散型データベースシステムである。密結合の単一連邦スキーマのシステムである。

B. Multibase

Multibase [SMIT81, LAND82] はCCA 社により開発されたシステムである。共通モデルとしては、関数型モデルであるDAPLEXがもちいられている。Multibase は、リレーショナルとネットワーク型のデータベースシステムを要素としている。密結合システムである。

C. Mermaid

Mermaid [TEMP87] は、Unisys社により開発された分散型データベースシステムである。4階層の構成を持ち、共通モデルはリレーショナルモデルである。外部層には、SQL と、一般利用者向けのARIEL とが提供されている。密結合システムである。

D. MRDSM

MRDSM(Multies Relational Data store Multidatabase) [LITW87] は、INRIA で開発されている多データベースシステムである。粗結合システムである。

E. PRESI *

PRESI * [DEEN85] は、Aberdeen大学で開発された分散型データベースシステムである。密結合システムである。

F. JDDBS

JDDBS [TAKI80, 83] は、共通モデルをリレーショナルモデルとし、4層の構造を持った分散型データベースシステムである。ネットワーク型データベースシステムを要素としている。

G. R*

R* は、IBM で開発されたリレーショナルの分散型DBMSである。

参考文献

[BATI86] Batini, C., Lenzerini, M., and Navathe, S., "A Comparative Analysis of Methodologies for Database Schema Integration," ACM Computing Surveys, Vol.18, No.4, 1986, pp.323-364.

[BERN87] Bernstein, P. A., Hadzilacos, V., and Goodman, N., "Concurrency Control and Recovery in Database Systems," Addison Wesley, 1987.

[BLAK87] Blakey, M., "Basis of a Partially Informed Distributed Database," Proc. of the 13th Intl. Conf. on VLDB, 1987, pp.381-388.

[CARD87] Cardenas, A., "Heterogeneous Distributed Database Management: The HD-DBMS," in [IEEE87], pp.588-600.

[CERI84] Ceri, S. and Pelagatti, G., "Distributed Databases - Principles and Systems," McGraw-Hill, 1984.

[CHEN86] Chen, P.P.-S., "The Entity-Relationship Model: Toward a Unified View of Data," ACM Trans. on Database Systems, Vol.1, No.1, 1976, pp.9-36.

[DAYA84] Dayal, U. and Hwang, H., "View Definition and Generalization for Database Integration in a Multibase Systems," IEEE Trans. on Soft. Eng., Vol.SE-10, No.6, 2984, pp.628-644.

[DEEN85] Deen, S. M., Amin, R. R., Ofori-Dwnmfuo, G. O., and Taylor, G. O., "The Architecture of a Generalized Distributed Database System - PRECI*," Computer Journal, Vol.28, No.3, 1985, pp.282-290.

[DEEN90] Proc. of the International Working Conf.. on Cooperating Knowledge Based Systems, (Deen, S. M. ed.), 1990.

[DU89] Du, W. and Elmagarmid, "Quasi Serializability: A Correctness Criterion for global Concurrency Control in Heterogeneous Database Systems," Proc. of the VLDB, 1989, pp.347-355.

[DURF89] Durfee, E.H., Lesser, V.R., and Corkill, D.D., "Trends in Cooperative Distributed Problem Solving," IEEE Trans. on Knowledge and Data Eng., Vol.1, No.1, 1990, pp.63-83.

[ELMA85] Elmasri, R., Weeldreyer, J., and Hevner, A., "The Category Concept: An Extension to Entity-Relationship Model," Data and Knowledge Engineering, North-Holland, 1985, pp.75-116.

[GARC87] Garcia-Molina, H. and Salem, K., "Sagas," Proc. of the ACM SIGMOD Conf., 1987, pp.249-259.

[HEIM85] Heimbigner, D. and McLeod, D., "A Federated Architecture for Information Management," ACM Trans. on Office Information Systems, Vol.3, No.3, 1985, pp.253-278.

[HSIO89] Hsiao, D. K. and Kamel, M. N., "Heterogeneous Databases: Proliferations, Issues, and Solutions," IEEE Trans. on Knowledge and Data Eng., Vol.1, No.1, 1990, pp.45-62.

[IEEE87] Proc. of the IEEE, Special Issue on Distributed Database System, 1987

[JACO88] Jacobson, G., et al., "CALIDA: A Knowledge-Based System for Integrating Multibase Heterogeneous Databases," Proc. of the

3rd Intl. Conf. on Data and Knowledge Bases, 1988, pp.3-18.
[KNAP87] Knapp, E., "Deadlock Detection in Distributed Databases," ACM Computing Surveys, Vol.19, No.4, 1987, pp.303-328.
[LAND82] Landers, T. and Rosenberg, R., "An Overview of Multibase," Distributed Databases (Schneider, H. -J. ed.), North-Holland, 1982, pp.153-184.
[LITW82] Litwin, W. et al., "SIRIUS Systems for Distributed Data Management," Distributed Data Bases (Schneider, H.-J. ed.), North-Holland, 1982, pp.311-366.
[LITW86] Litwin, W. and Abdellatif, A., "Multidatabase Interoperability," IEEE Computer, No.12, 1986, pp.10-18.
[LITW87] Litwin, W. and Abdellatif, A., "An Overview of the Multidatabase Manipulation Language MDSL," Proc. of the IEEE, Special Issue on Distributed Database System, 1987, pp.621-632.
[MOTR81] Motro, A. and Buneman, P., "Constructing Superviews," Proc. of the ACM SIGMOD Conf., 1981, pp.54-64.
[NAVA86] Navathe, S., Elmasri, A., and Larson, J., "Integrating User Views in Database Design," IEEE Computer, Vol.19, No.1, 1986, pp.50-62.
[ONUE83] Onuegbe, E., Rahmimi, S., and Hevner, A., "Local Query Translation and Optimization in a Distributed System," Proc. of the AFIP Conf., 1983, pp.229-239.
[OSI] "Data Processing - Open Systems Interconnection - Basic Reference Model," ISO 7498, 1987.
[RAM89] Ram, S. and Chastain, C., "Architecture of Distributed Data Base Systems," Journal of Systems and Software, Vol.10, No.2, 1989, pp.77-95.
[RUSI89] Rusinkiewicz, M., et al., "OMNIBASE: Design and Implementation of a Multibase System," Proc. of the 1st Annual Symp. in Parallel and Distributed Processing, 1989, pp.162-169.
[SHET88] Sheth, A., "Managing and Integrating Unstructured and Structured Data: Problems of Representation, Features, and Abstraction," Proc. of the 4th Intl. Conf. on Data Eng., 1988, pp.598-599.
[SHET90] Sheth, A. and Larson, J.A., "Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases," ACM Computing Surveys, Vol.22, No.3, 1990, pp.183-235.
[SMIT81] Smith, J. M., Bernstein, P. A., et al., "Multibase - Integrating Heterogeneous Distributed Database Systems," Proc. of AFIPS, 1981, pp.487-499.
[TAKI80] Takizawa, M. and Hamanaka, E., "Query Translation in Distributed Databases," Proc. of the IFIP'80, 1980, pp.451-456.
[TAKI83] Takizawa, M., "Distributed Database System - JDDBS," JARECT Vol.7, Computer Science and Technologies (Kitagawa, T., ed.), Ohmsha and North-Holland, 1983, pp.262-283.
[TAKI90] Takizawa, M. and Deen, S. M., "Synchronization Problems of Compensate Operations in the Object-Model," Proc. of Interna-

tional Working Conf. on Cooperating Knowledge Based Systems, Keele, England, 1990.

[TEMP87] Templeton, M., et al., "Mermaid - A Front-end to Distributed Heterogeneous Databases," in [IEEE87], pp.695-708.

[TSIC78] Tsichritzis, D. and Klug, A., "The ANSI/X3/SPARC DBMS Framework," Information Systems, Vol.3, No.4, 1978.

[VEIJ86] Veijalainen, J. and Popescu-Zeletin, R., "On Multidatabase Transactions in a Cooperative, Autonomous Environment," TR., Hahn-Meitner Inst., Berlin, 1986.

[WEIH89] Weihl, W. E., "Local Atomicity Properties: Modular Concurrency Control for Abstract Data Types," ACM Trans. on Programming Language and Systems, Vol.11, No.2, 1989, pp.249-283.

[YU84] Yu, C. and Chang. C., "Distributed Query Processing," ACM Computing Surveys, Vol.16, No.4, 1984, pp.399-433.

[ZANI79] Zaniolo, C., "Design of Relational Views Over Network Schemas," Proc.of the ACM SIGMOD Conf., 1979, pp.179-190.

3．グループウェアデータベースシステム

従来のリレーションモデルやネットワーク型のモデルのデータベースシステムは、事務処理等の定型的な業務には適しているが、グループワークといった複数の利用者が共同して行う作業を支援することには適していない。ここでは、図形、イメージといったマルチメディア情報、ソフトウェアモジュール、スケジュールといった種々の情報が必要となり、これらの相互の関連構造を扱えるデータベースシステムが必要となる。このためには、以下の技術が必要となる。

3.1　オブジェクト指向データベースシステム

オブジェクト指向システムでは、データ構造とこれを操作するための演算（手続き）を一体化したオブジェクトを基本としている。グループワークでは、グループの構成員が仕事を行うときに、作業を他に依頼する。ここで、依頼されるものをオブジェクトとしてとらえることができる。

3.2　分散型データベースシステム

グループワークでは、種々のデータベースシステムを利用する必要がある。データベースシステムとしては、グループ全体で共用されるものと、各個人のデータベースシステムとの二種を考える必要がある。グループワークでは、グループ全体で共用される共用データベースシステムと、各個人で管理される個人データベースシステムとの協調が必要となる。また、各個人は、一つのグループワークに参加するだけでなく、種々のグループワークに参加する。例えば、あるソフトウェアの開発プロジェクトがあるとともに、個人的に古典の研究グループに参加している。このように、グループワークでは、各個人の持つ情報をどのように管理していくかが主要な問題となる。

3.3　利用者インタフェース

グループワークを行うためには、利用者インタフェースが重要である。利用者インタフェースは、利用者が日常の生活の中で、行っているものと同じ形でデータベースシステムを操作できることが必要である。例えば、ある情報を検索するとき、検索用のコマンドを入力するのではなく、日常行われるように本を調べるのと同じように、本をめくりながら検索を行えることが望ましい。このために、メタファの概念は有効である。グループメタファは、グループワークをあたかも同じ場所で働いているのと同じように操作を行わせるものである。

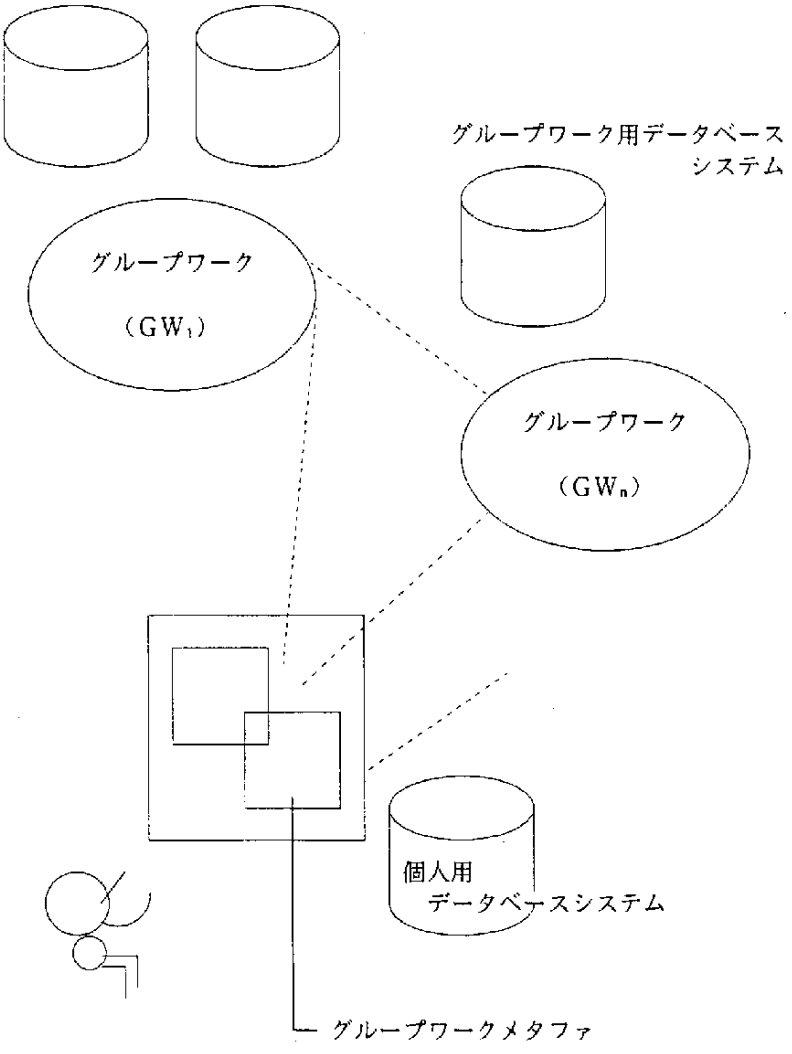


図2-1　グループウェアデータベース

Ⅲ. グループウェア発展に伴う情報システムの高度化ならびにデータベース構築技術の進展とその課題

1．グループウェアの発展と情報システム高度化

コンピュータ技術とネットワーク技術の進展により、現在は、高度情報化社会といわれる。しかし、一方では、東京を中心とした首都圏への一局集中化によるオフィス空間の確保が困難となり、また通勤地獄といわれるような長時間の勤務等の労働環境の悪化が問題となってきている。また、高性能、高機能なロボットの導入、QCサークル等により、欧米の先進諸国に対しても、生産現場での生産性が向上し、安価で品質のよい製品を生産できるようになっている。これに対して、事務分野では、依然として労働集約型であり、多くの人間による手作業的な仕事と、会議、打ち合わせに多くの時間がとられ、その生産性の向上が求められている。

今後の情報化社会では、ローカルエリアネットワークに加えて、光ケーブルを用いたBISDN 等によるGbpsオーダの超高速ネットワークが社会のすみずみまで普及した形となり、また、放送と電話が一体化した通信が可能となってくる。こうした情報化社会では、会議や打ち合わせ等の共同作業を行うために、人間が移動し、作業するのではなく、必要な情報を各個人に移動させることが必要となる。人間が場所を移動し、仕事を行うことから、情報を移動させるだけの通信帯域を利用できる時代となってきている。

これらに対する一つの解として、オフィスを分散されることがある。分散オフィス（サテライトオフィス）とは、部、課、支店といった組織単位が分散するのではなく、個人単位に、常勤して分散勤務するオフィスを意味する。オフィスでの主要なワークとして、構成員の個人ワークとともに、複数の構成員によるグループワークがある。こうしたグループワークをどのように支援していくかがグループウェアの課題である。分散オフィスを実現するためには、複数のサイトに分散したグループ構成員によるグループワークをどのように支援していくかが問題となる。基本的に、オフィスでのワークは、種々の情報に基づいて行われる。これらの情報には、伝票、住所録、電話帳のような定型化されたものから、個人間で通信される顔色、雰囲気、気持ちといった非定型なものまで、種々のものがある。これらの情報は、データベースシステムとして計算機内に記憶され

るならば、互いに共用することにより、種々の処理を行うことができる。このように、分散オフィスを実現するためには、オフィスでのグループワークを行うための情報を管理するためのデータベースシステムが重要なものとなる。

2．グループウェアとデータベース構築における今後の課題

1981年に、Johnson-Lenz夫妻によって始めてグループウェアという用語が使われ、また、1986年にはIrene Greif らによりグループウェアを専門的に研究するCSCW(Compute Supported Cooperative Work ）と呼ばれる新しい学際的な研究分野が開択されてきた。また、最近では、グループウェアに対する急激な関心と期待が高まってきている。しかし、こうした期待に応えいくためには、解決すべき多く問題点や更に研究すべき課題が数多く残されている。ここでは、それらの問題のいくつかについて考えてみる。

2.1　グループウェア設計上の問題点

今迄に数多くのグループウェアが開発されてきた。しかし、これらの製品は、必ずしも組織に受け入れられたり、期待された効果を上げていない。その原因としてJonathan Grudinは、次の三つの点を指摘している［Grudin］。

①　グループウェアを受益者と、そのために必要な作業を行う非受益者

グループウェアが機能するためには、グループの全員がグループのために何らかの作業を行う必要がある。例えば、簡単な例としてグループのスケジュール調整システムがある。このシステムは、グループメンバーのスケジュールをチェックし、会議に全員が参加できる日時等を調べたり、その日を通知するものである。しかし、こうしたシステムの受益者は、管理者であり、多の多くのメンバーは、そのために常に最新のスケジュールをデータベースの登録するという作業が強制されるだけであり、組織に受け入れられにくい。

②　アプリケーション開発の直感的動機

スケジュール調整システムの様なアプリケーションが、何故できてしまうかというと、開発決定者は、ワードプロセッサーのような単一ユーザ向けのアプリケーションと同じ

感覚でグループウェアの設計を行うからである。その結果、グループウェアが色々なバックグラウンド持った人や、異なった職種の人の参加によって成り立つという観点が抜けてしまう。開発決定者は、一部の人（多くは管理者）に利益がもたらされるアプリケーションを開発したがる。管理者に作業負担がかかるようなグループウェアの開発は行わない。

③　評価の困難さ

グループウェアの評価は、単一ユーザ向けのアプリケーションに比べて、極めて困難でほとんど不可能である。それは、グループウェアを使うグループの構成は多種多様で同じでないからである。従って、他での経験が利用できない。

2.2　グループウェア利用上の問題点

最も単純なグループウェアとして電子メールがある（ただし、電子メールをグループウェアと呼ばない場合もある）。電子メールは、コンピュータリソースの共用の目指したDARPA プロジェクトで開発されたARPANET と呼ばれるネットワーク上で当初付加的に開発されたものであるが、時間と空間を越えて簡単に研究者が通信できることから急速に普及し、最も利用されるサービスとなった。現在では、各種の商用パッケージや電子メールサービスを提供する機関が存在し異なった組織間とのやり取りも容易になってきている。

しかし、電子メールのような簡単なアプリケーションでも組織に導入するには、

①　全員が使えるようにはならない

②　効果が明確でない

③　PC等のコンピュータやネットワークの設置が必要になる

④　他人が簡単に読んでしまうなどのプライバシ侵害がある

⑤　公用と利用の区別がはっきりしない

等の問題がある。

多くのグループウェアは、電子メールをインフラストラクチャーとしており、まず、これらの問題をクリターする必要がある。

2.3　グループウェアの統合化

既に数多くのグループウェアが研究開発され、その一部は製品化されている。しかし、これらは何れも個別に開発されてきているため互いに孤立したものとなっている。例えば、多くのグループウェアと呼ばれる製品は、基本的に電子メールに機能をもち、これと併せてスケジュール管理機能、会議室予約機能、協同執筆機能、ワークフロー機能等をもっている。これらの多くは、共通点をもち、また利用者は、組織の目的に合わせてそれらを組み合わせて利用したとの要求をもっている。

MOCCA では、既存のグループウェアやそこで用いられているモデルを分析し、図に示すCSCW統合化のためのアーキテクチャーを開発した。

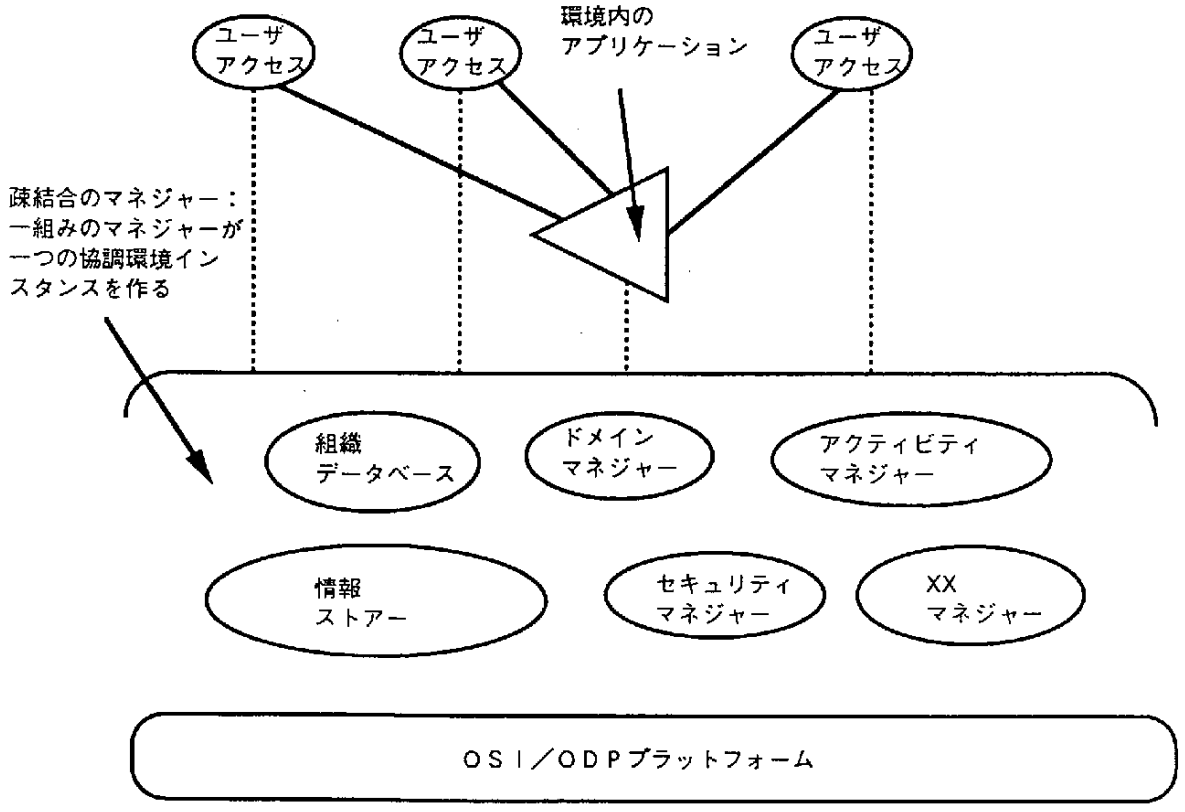


図　MOCCA アーキテクチャー（出典：[Mocca]）

MOCCA は、IOS で標準化されているOSI （開放型システム相互接続）や標準化が進められているODP （開放型分散処理）を基本的なコミュニケーション基盤としている。
この上に現在のところ次の５つのマネジャーを考て、各種アプリケーションが関連をもって実行できるようにしている。

① ドメインマネジャー

ドメインマネジャーは、各種オブジェクトをドメイン単位にまとめ見通しのようものにするためのものである。ドメインとは、ポリシーあるいは共同作業の単位をいう。

② アクティビティマネジャー

アクティビティマネジャーは、アプリケーションで支援されるアクティビティを登録する。例えば、アクティビティの目的、状態、参加者、リソースなどがアクティビティマネジャーに登録される。

③ 情報ストア

情報ストアは、環境内に存在するオブジェクトに対する共通的な蓄積サービスを提供する。

④ 組織データベース

組織データベースは、組織に関する情報が蓄積する。例えば、組織で使われるリソース、従業員、役割、組織会の関連、組織の手順やポリシーが蓄積される。

⑤ セキュリティマネジャー

セキュリティマネジャーは、システム内のリソースの利用やアクセスを管理する。
この他にも、例えばMERMAID 等が用いているアプリケーションの複型を管理するマネジャーの必要性についても検討している。
ISO のMOTS (Message Oriented Text Interchange System) やCCITT のMHS （Message Handling Systems）では、グループウェア構築上の重要な技術であるメッセージの構造化が検討されてない。今後は、OSI やODP のメンバーとCSCWのメンバーが一堂に会して全体的な統合アーキテクチャーの検討を行っていくことが望まれる。

2.4 協調理論の発展

グループウェアを開発していくためには、人間がいかに協同して作業を進めていくかに

ついての深い理解とそのモデル化が必要となる。

これらの協調に関する研究は、今迄コンピュータサイエンス、経済学、政治学、オペレーションリサーチ、組織理論、生物学、心理学等の世界でバラバラに研究されてきた。しかし、問題が複雑であることから今後は認知科学のような学際的な研究が必要となる。認知科学は、それまで異なったフィールドで別々に研究されていた問題に共通点があることの認識からスタートした学問である[Malone]。

参考文献

[Grudin] Jonathan Grudin : Why CSCW Application Fail : Problems In The Desogn And Evaluation Of Organization Interfaces, CSCW'88 Proceedings, pp.85-93 (1988).

[Mocca] The Mocca Group : Mocca : An Environment for CSCW Applications, ACM SIGOIS Bulletin , pp.21-23 (1992.4).

[Malone] Thomas W. Malone et al. : Toward an Interdisciplinary Theory of Coodination, Center for Coodination Science MIT (1992).

平成5年3月発行

発　行　財団法人　データベース振興センター
東京都港区浜松町二丁目4番1号
世界貿易センタービル7階
TEL　03（3459）8581

委託先　株式会社　イフ・アドバタイジング
東京都新宿区四谷4-21-37
TEL　03（3355）5371

印　刷　株式会社　エーヴィスシステムズ
東京都文京区本郷2-39-5
TEL　03（3816）4111