マルチメディア通信と分散処理 52－10
(1991. 9. 25)

並行協調作業の支援に適した多重レイヤ構造の提案

塚田 晃司, 市村 哲, 岡田 謙一, 松下 温

慶應義塾大学

実社会における作業形態を見ると、作業者一人のみで作業を進めることは稀であり、「グループ」により協調して作業を進めている。一方、個人の作業形態は、一つの作業のみに作業時間を占有されることはなく、複数の作業に同時に携わっていたり、あるいは、複数のグループに同時に属して作業を進めている。

以上のような「並行協調作業」形態に対して、従来の支援形態では、どの作業をする場合でも同一の作業環境の中で作業せざるを得ず、複数の作業を処理するにあたって作業環境が画一的であることは、必ずしも効率的であるとは言い難い。

そこで、並行協調作業に適した「多重レイヤ構造」を提案する。また、多重レイヤ構造を採用したデータベースの実験システムを試作したので報告する。

A Multi-layerd Architecture Suitable for Concurrent Collaborative Work

Koji TSUKADA, Satoshi ICHIMURA,
Ken-ichi OKADA, Yutaka MATSUSHITA

Keio University

3-14-1 Hiyoshi, Kohoku-ku, Yokohama 223, Japan

In paying attention to activities of actual organizations, the activities are generally based on "concurrent collaborative work" in which each worker promote several kinds of work concurrently. However, in former computer supported environments, workers could not but work in an uniform work environment regardless of the kinds of work. Therefore, we aim to provide the computer supported environment in which only the necessary information for current task is available, and we developed an experimental database management system which employs a "multi-layerd architecture".

1 はじめに

従来、コンピュータによる作業支援の形態は、個人環境の支援に重点が置かれていた。しかし、実社会における作業形態を見ると、作業者一人のみで作業を進めることは稀であり、ほとんどの場合、複数の作業者から構成される「グループ」により協調して作業を進めているのが実状である。

一方、個人の作業形態に注目すると、一つの作業のみに作業時間を占有されることはなく、複数の作業に同時に携わっていたり、あるいは、複数のグループに同時に属して並行して作業を進める場合が多々ある。

以上のような作業形態に対して、従来の支援形態では、どの作業をする場合でも同一の作業環境の中で作業せざるを得なかった。特に、一人に一台のワークステーションが与えられるような環境においては、複数の作業を処理するにあたって、作業環境が画一的であることは、必ずしも効率的であるとは言い難い。

そこで、本研究では、上述のような作業環境の実状を鑑み、並行協調作業に適した「多重レイヤ構造」を提案する。また、多重レイヤ構造を採用したデータベースの実験システムを試作したので報告する。

2 並行協調作業

実社会においては、上述のように、個人が複数の作業をグループで協調しながら、並行的に進めるのが一般的である。例えば、図1のような場合が考えられる。

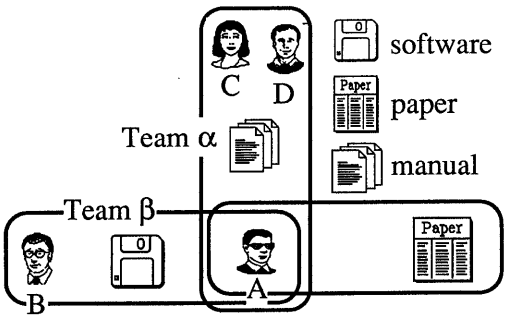

図 1: 並行協調作業

この例では、作業者Aは作業者Bと協同してあるソフトウェアを開発するのと同時に、作業者C、Dとともにマニュアル作成に携わっており、また、独自に自分の研究に関して論文を執筆している。

このような場合、従来の作業環境が画一的な支援形態では、どの作業をする場合でも同一の作業環境の中で作業せざるを得ない。特に、一人に一台のワークステーションが割り当てられる環境が整いつつある昨今、種々の作業の処理にあたって、各作業に適した作業環境を提供できない支援形態では必ずしも効率的に作業できるとは言い難い。

さらに、このような環境の下では、作業者は現在作業中の作業の種類に関係なく、常に自分のワークステーションの管理下にあるすべての(参照可能な)情報を参照することが可能である。そのため、どの情報がどの作業に必要であるか、すなわち、情報と作業の関連性については各作業者が独自に管理しなければならない。

したがって、従来の支援形態により並行協調作業を支援する場合には、複数の作業に必要な情報を冗長して保存していたり、グループで協調して作業する場合には、共通に必要な同一の情報を各メンバが独自に所有しているなど、情報管理において非効率的な状態が生じ得る。

一方、昨今、コンピュータ、および、その周辺環境は急速に向上し、また、光磁気ディスクなどの大容量で高速な二次記憶装置の入手も容易である。さらに、光ファイバなどによる高速なネットワーク環境も整いつつあり、その結果、莫大な量の情報の参照、処理が可能となっている。

このような現状に対して、人間の方は処理できる情報量に限界があるため、コンピュータからの情報のすべてを処理することができず、必要としている情報が不要な情報に埋もれてしまう状況が起こり得る。そのため、莫大な量の情報の中から必要としている情報のみを選択して取り出す「情報のフィルタリング」の必要性が論じられている[Greif 87]。

3 多重レイヤ構造

そこで、本研究では、情報の非効率的な管理、および、情報の爆発的な増加を防ぐためには、必要としている情報のみが利用可能で、不要な情報は隠蔽されるべきであると考えた。そして、情報は

- 情報を個人に所有されるべき情報

- グループのメンバ間、あるいは、幾つかのグループ間で共有されるべき情報

の二種類に分類できると考えた [塚田 91]。

ただし、共有される情報は、当該グループのメンバならばあたかも個人で所有している情報のごとく透過して参照することができるものとする [市村 90]。

このように、必要としている情報のみが利用可能で、共有すべき情報は透過して扱うことができれば、冗長に情報を保存する必要がなくなる。さらに、複数の作業に対する情報と、その作業環境を統合することが可能となる。

本研究では、この「情報の透過的共有」を実現するために「多重レイヤ構造」を提案する。多重レイヤ構造について具体的に述べると、各個人、あるいは、個人の作業環境毎に独立したレイヤ (以後、「個人用レイヤ」と呼ぶ) が割り当てられ、その他に透過的共有される情報を管理するレイヤ (以後、「共用レイヤ」と呼ぶ) が存在する (図 2 参照)。

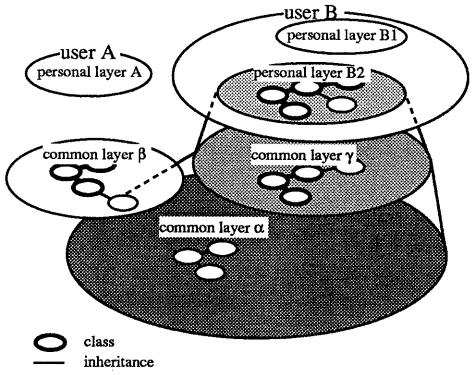

図 2: レイヤ構造

個人は、各々の個人用レイヤ上の情報に対して自由な新規作成、変更、および、参照が許可されている。しかし、共用レイヤ上の情報に対しては、共用レイヤの管理者である場合を除いて、原則的に参照のみが許可されるものとする。

一方、共用レイヤは、個人用レイヤ、あるいは、当該レイヤより上位の共用レイヤの下位レイヤとして存在する。そして、各レイヤは、当該レイヤより下位のすべての共用レイヤ上の情報をあたかも当該レイヤ上の情報のごとく透過して扱うことができる。ただし、透過された情報は、上述のように、原則的に参照のみ許可されるものとする。

以上に述べた多重レイヤ構造により、作業者は、各個人、あるいは、各作業毎に独立したレイヤ上での作業が可能となる。また、グループでの作業に必要な共通の情報は、共用レイヤ上に配置することによって各個人用レイヤに透過して扱うことが可能となり、また、冗長に情報を保存する必要がなくなり、必要とする情報のみが参照可能となる。

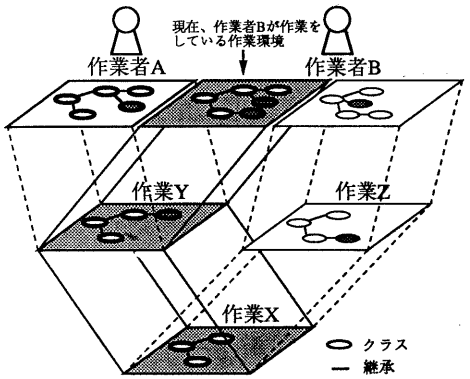

図 3: 情報の透過的共有

図 3を例にとると、作業者 A は作業 Y を担当し、作業者 B は作業 Y 、および、作業 Z を担当している。また、作業 Y 、および、作業 Z は、作業 X の一部分である。

現在、作業者 B は作業 Y に従事しており、作業者 B の作業環境は、作業 Y の共用レイヤを透過して、作業者 A と情報を共有している。更に、作業 Y の共用レイヤは、作業 Z の共用レイヤと共に、作業 X の共用レイヤ上の情報を共有している。

したがって、多重レイヤ構造により作業に対する情報とその作業環境を統合することができる。その結果、「作業を選択することによって、それに適した作業環境が得られる」という「作業選択型」の環境が実現される。

4 実験システム

我々は、以上の多重レイヤ構造を採用し、オブジェクト指向データモデルにも対応した分散データベースの実験システムを LAN により接続され

たSUNワークステーション上で作成した。

本実験システムの構成について概説すると、分散環境下の各ワークステーション上に一つプロセスとしてのデータベースサーバが存在する。それらが各々一つの個人用レイヤ、あるいは、共用レイヤに対応している。

その他のプロセスとして、幾つかのユーザインタフェースクライアントが存在し、作業者とデータベースサーバの間のインタフェースを司る（ 図4参照 ）。

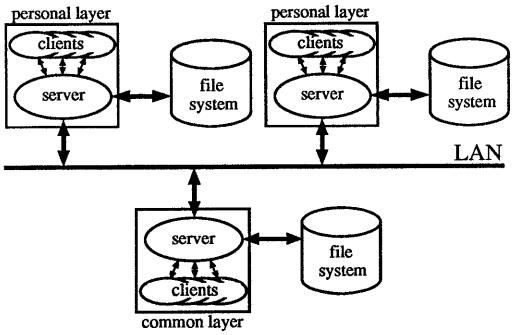

図 4: 実験システム構成の概要

個人用レイヤ上の情報を参照する場合には、当該個人用レイヤのユーザインタフェースクライアント、データベースサーバ間の通信により実現される。

一方、共用レイヤ上の情報を透過して参照する場合には、LAN を介した当該レイヤ間のデータベースサーバ同士の通信により実現される。

このようにシステム内部では、参照方法が情報の物理的位置により異なるが、作業者は参照方法の違いについて意識する必要はなく、個人で所有している情報と同様に参照することが可能となっている。

以下で、もう少し詳しく本実験システムの実現に関して触れる。

4.1 多重レイヤの実現

上述のように、各レイヤは、実際には各データベースサーバに対応している。原則的に一つのサイト内には、アクティブなレイヤが一つのみ存在するものとする。

データベースサーバの役割は、当該レイヤのユーザインタフェースクライアントからの要求、および、他のレイヤのデータベースサーバからの要求の受け付け、当該レイヤのデータに対して、参照、変更、検索などの処理、そして、処理結果を返答である。

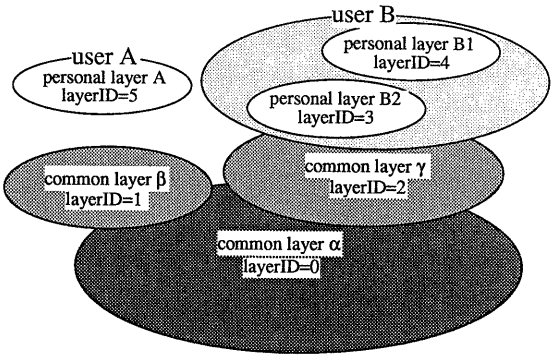

図 5: レイヤ ID

各レイヤには、レイヤを識別するためにレイヤ ID が割り当てられている。この ID は、LAN により接続された分散環境内で一意に決定できるものでなければならない (図5 参照)。

なぜならば、レイヤ ID は、レイヤ間の通信、データ管理の際に重要な役割をもち、レイヤのアドレス、データの透過、および、非透過に関する情報を含んでいるからである。

4.2 オブジェクト指向の実現

オブジェクト指向データモデルに対応するために、本実験システムでは、データ管理の際にクラス階層構造を採用している。さらに、本システムでは、クラスも透過的共有することが可能となっている。

クラスは、大きく

- システム固有のクラス
- ユーザ定義によるクラス

の二つに分類できる。

前者には、Object 、Int 、Double 、String 、Text 、PilotCard 、Set などがある。その有効範囲はシステム全体に及ぶ。

一方、後者はユーザが独自に定義することが可能なものであり、これらの有効範囲はレイヤの透過性からそのクラスが定義されているレイヤ、および、当該レイヤより上位のレイヤのみである。これにより、各ユーザは既製の形にとらわれずに独自のデータ管理が可能となる。

本来、システム固有のクラスは、システム全体に共通であるので多重レイヤの最も下位のシステムの基盤となるレイヤ上にそのクラス定義が存在し、他のレイヤは、その定義を透過して扱うことが理想である。

しかし、透過しているクラス定義を参照する度に当該レイヤと通信するオーバーヘッドを避けるために、透過しているクラス定義の複製を各レイヤで保持している。したがって、実際には、各レイヤ上には、すべてのクラス定義が存在することになる。

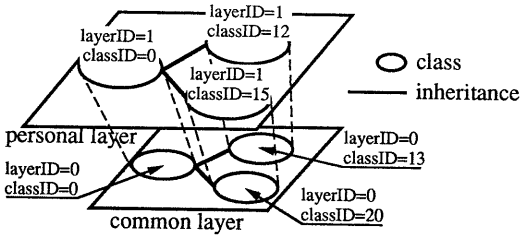

図 6: クラス ID

本システムでは、クラスを管理するために各クラスにクラス ID を割り当てている。この ID は、各レイヤ内で一意に決まるものである (図 6 参照)。

以上のことから、透過しているクラスの ID は、そのレイヤの下位のレイヤにおける ID と必ずしも同一ではない。そこで、透過しているクラスは、下位のレイヤでの当該クラスの ID の情報を保持し、対応関係を記述している。

以上のことから、クラスは、先に述べたレイヤ ID とクラス ID の組によりシステム全体で一意に決定できることになる。このことを利用して、本実験システムでは、UNIX のファイルシステム上で、レイヤ ID とクラス ID の組でクラスを管理している。

各インスタンスには、オブジェクト ID を割り当てている。この ID は、同じオブジェクトに対

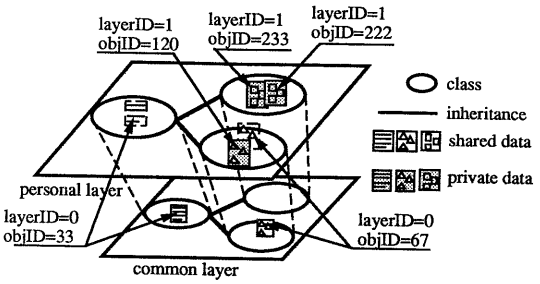

図 7: オブジェクト ID

しては同じ ID を割り当てられ、各レイヤ内で一意に決まるものである (図 7 参照)。

したがって、あるオブジェクトが複数のクラスのインスタンスであってもインスタンス ID は唯一である。

以上のことから、オブジェクトは、レイヤ ID とインスタンス ID の組によりシステム全体で一意に決定できることになる。このことを利用して、クラス管理の場合と同様に、UNIX のファイルシステム上で管理している。

4.3 分散環境での実現

分散環境で実現するために本実験システムでは、UNIX のシステムコールである socket[Sun 90A]、および、message[Sun 90B] を利用している。これらを適宜組み合わせて、分散環境下でのサーバクライアントモデル [Svobodova 84] を実現している。

したがって、システム内の通信は、

- サーバ↔サーバ間通信
- サーバ↔クライアント間通信

に分類できる。

すると、本システムのシステム構成 (図 4 参照) では、前者は、ネットワークを介したプロセス間通信であるのに対して、後者は、同一サイト内、すなわち、ネットワークを介さないプロセス間通信であり通信手段が異なる。本システムではこの差異を吸収するために、前述のソケット、および、メッセージを適宜組み合わせて利用している。

具体的には、各レイヤ毎に要求用キューと返答用キューを用意し、要求、および、返答の通信は、

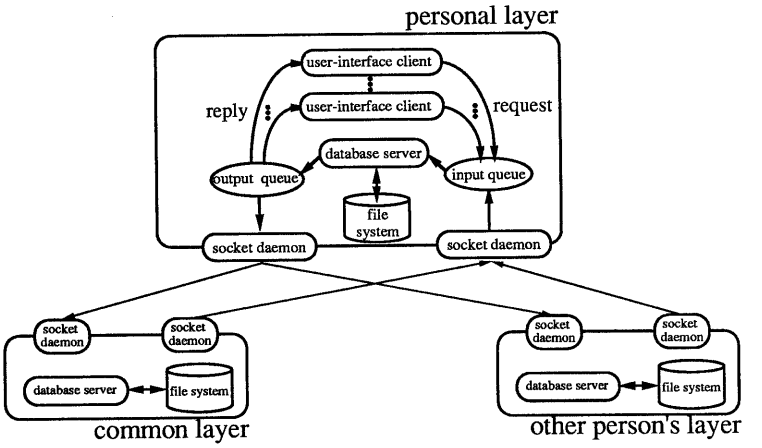

図 8: プロセス間通信

一度、各キューにメッセージを用いて蓄えられる。その後、他のレイヤへの通信、レイヤ内の通信に分類され、他のレイヤへの通信ならばソケットを用いて当該レイヤに送信される (図 8 参照) 。

以上の処理は専用のプロセスにより実行されているので、サーバプロセス、および、クライアントプロセスは、通信手段の差異を意識することなく通信することが可能となっている。

また、透過的共有されている情報への要求に対しては、その情報が実在するレイヤのデータベースサーバに至るまで、要求が上位のレイヤから順次、下位の当該レイヤまで転送される。しかし、それに対する返答は、直接、要求を出したレイヤに送られる。

5 おわりに

本研究で、我々は、実社会の作業形態に即した「並行協調作業形態」を支援するのに適した「多重レイヤ構造」を提案し、また、実験システムを試作した。これにより、従来の支援形態では対応できなかった「作業本位の環境」が実現されたといえるだろう。

参考文献

[Greif 87] I. Greif and S. Sarin, "Data Sharing in Group Work", ACM Transactions on Office Information Systems, Vol.5, No.2, 1987.

[市村 90] 市村, 松浦, 岡田, 松下,「分散協調型作業支援システム—チームウェア」, 情報処理学会 1990 年代の分散処理シンポジウム論文集, 1990.

[Svobodova 84] L. Svobodova, "File Servers for Network-Based Distributed Systems", ACM Computing Surveys, Vol.16, No.4, 1984.

[塚田 91] 塚田, 市村, 松浦, 岡田, 松下,「並行協調作業支援のための多重レイヤデータベース」, 情報処理学会第 42 回全国大会, 1991.

[Sun 90A] Sun microsystems, "A Socket-Based Interprocess Communications Tutorial", Network Programming Guide, pp.251–278, 1990.

[Sun 90B] Sun microsystems, "System V Interprocess Communication Facilities" Programming Utilities & Libraries, pp.53–92, 1990.