Vol. 37 No. 12 情報処理学会論文誌 Dec. 1996

分散オブジェクト指向による作業融合支援プラットフォーム FusionWorks の開発

木本陽介[†] 服部進実[†]

本論文では，著者らが開発した分散オブジェクト指向による作業融合支援プラットフォーム“FusionWorks”について述べる．本システムは従来の考えに基づくグループウェアの支援対象を拡張し，作業結果がまとめられる過程も支援対象とすることで，作業者に対し柔軟な作業空間を提供するものである．その特徴は，コミュニケーション機能と作業支援機能の分離，また，作業をイベントとすることで，作業過程に包含される情報を具体的に再利用できることの2つにある．これらを実現するために，作業支援機能のプラットフォーム化と作業者の発言を再利用可能なイベントに変換する新たな通信システムを開発し，その有効性を確認するために会議支援システムへの応用を行った．

Development of Integrated Work Support Platform “FusionWorks” Based on Distributed Object Orientaion

YOUSUKE KIMOTO[†] and SHIMMI HATTORI[†]

In this paper, we describe globally integrated work support platform “FusionWorks” which can realize flexible work circumstances to user, extending targets of support from conventional groupware objects to integrated process of them. Main characteristics of this system are separation of communication and work support function, and reusability of work process information by making work event. In order to realize these characteristics, integrated work support platform and communication system which can convert worker's statement to reusable event have been developed, being applied conference support system to confirm the availability.

1. はじめに

近年コンピュータのオフィス業務利用は，ワードプロセッサ，表計算，CAD等多様化し，個人作業空間だけではなく，組織，グループといった集団で行う協調作業空間としてその利用が進んでいる．このような急速な業務利用の普及には，従来までの特化されたシステムが，ウインドウシステムによるGUIプラットフォームの普及によって汎用化されたことが大きく貢献している．しかし，現在のウインドウシステムのアプリケーションは利用環境がGUI化されただけで，従来までの使用目的によるシステムの特化と大きな変化はないと考えられる[13]．

このような状況において作業支援を行うシステムとしてグループウェアがある[11),15]．ユーザは作業目的別に統合されたアプリケーションを利用することで，効率的に作業を進めることが可能である．しかし，グループウェアですべての作業を支援することは不可能であり，ユーザはアプリケーションごとにその操作方法を学習する必要があるため，複数の作業を並行して進めることが困難である．

また，一方で作業におけるデータ再利用の点から，最近コンポーネントウェア[18]が注目されている．コンポーネントウェアではテキストや図形などをまとめ，複合文書として利用可能とするものである．しかし，コンポーネントウェアにおいては表オブジェクトや図形オブジェクトなどアプリケーション固有のデータを結果として利用するにとどまっており，協調作業において重要な作業過程の再利用はできない．

以上の問題を解決するために，作業支援システムとしての新たな考えに基づくプラットフォームの構築が必要と考えた．プラットフォームはユーザからの作業要求に柔軟に対応するために，使用するアプリケーションの応用能力，作業の再利用性，またネットワーク利用のためのコミュニケーション機能の充実を目的

† 金沢工業大学工学部情報工学科
Department of Information and Computer Engineering, Kanazawa Institute of Technology

としたものである．

本論文では，作業支援システムに必要な機能，作業の再利用性について考察を行い，新たなプラットフォームの構築方法，またそれを利用したアプリケーションを通してプラットフォームの有効性について述べる．

2. 柔軟な拡張性を持つ作業支援環境

現在グループウェアでは，映像や音声の送信，メールの管理，メールにボタンなどを付加した多機能メールの実現など，分散環境における簡単なコミュニケーション支援にとどまっていると考えられる．しかし，グループウェアの目的はコンピュータを利用しての作業支援であり，現在のグループウェアはその点に関して不十分であると考えられる．本章では作業支援環境に必要とされる機能について述べる．

2.1 作業支援環境の要求条件

グループウェアを利用すると，協調作業を実現するための専用の機能が付加された共有ホワイトボードや，KJ 法ツール[12]などの利用が可能となる．分散環境での作業共有を実現するうえで，このような共有空間を提供するツールは有効であるが，その機能がグループウェアアプリケーション内に限られており，作業支援機能が特化されている点が問題である．また，この機能特化によって，グループウェアを利用しての作業結果は共有空間に依存した存在となる．このため，協調作業終了後ユーザがその結果を個人的な作業に再利用する，といった作業効率の向上にはつながらないと考えられる．

このような問題の背景には，グループウェアが提供する作業空間に，ユーザを強く束縛したことが大きく関係していると考えられる．この束縛によってユーザの作業空間における自由度が限定されると同時に，生成される作業結果の再利用性も低下すると考えられる．

そこで，グループウェアにおけるユーザの自由度を向上させるために，その基本機能と利用可能なアプリケーションの分離を行うことが必要である．つまり，目的とする機能は共有ホワイトボードや，KJ 法ツールなどではなく，共有空間を実現するコミュニケーション機能を提供することにあり，共有機能を包含した個々のアプリケーションごとに共有機能を提供するのは，冗長性が生まれ無駄であると考えられるためである．また，アプリケーションは，ユーザが個人的な作業で利用している使い慣れたアプリケーションを利用することが必要である．これは，アプリケーションを分離することが単に機能の役割分担ではなく，アプリケーションの利用しやすさを向上させることが必要なため

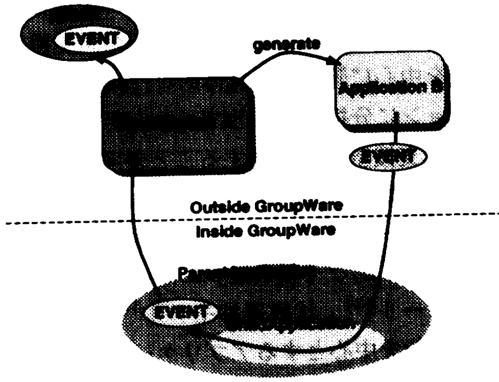


図1 アプリケーションの依存関係
Fig. 1 Dependency of applications.

である．

以上からグループウェアの基本機能を提供する新たなプラットフォームと，個人作業空間，協調作業空間を区別しないアプリケーションの提供が必要である．

2.2 アプリケーション相互依存関係の必要性

前節で述べたように，グループウェアの基本機能を分離することによって，アプリケーションの独立性が保証され，ユーザの使い慣れたアプリケーションを協調作業空間に反映することが可能となる．しかし，このアプリケーションの独立だけでは，アイデアをまとめ，具体的に実現するなど機能別に作業が進められることとなり，作業間の連係はユーザ側での管理を必要とする．つまり，協調作業において作業目的に応じてアプリケーションを複数起動した場合，その起動されたアプリケーションを独立して管理を行うのでは，協調作業に必要とされる発言や作業の流れの管理が困難であることを示している．よって，このような問題を解決するために，起動されたアプリケーションをすべて独立して管理するのではなく，アプリケーションに依存関係を付加する機能が必要である．

この依存関係は，あるアプリケーションが起動された時，そのアプリケーションから新たなアプリケーションが起動される場合に発生する関係である．図1 は Application A から Application B が生成され，親，子の依存関係の発生を示している．このとき，Application A を親アプリケーション，Application B を子アプリケーションと定義する．依存関係によって子からの発言は親へ渡され，親が代理転送を実行し 図1 の親のイベント（イベントについては 2.3 節で後述する）の内部に子のイベントを包含した形が実現される．このような独立したアプリケーション間の依存関係によって，複数のアプリケーションを利用した，協調作業における発言の流れを管理することが可能となる．

また，この依存性によってグループウェア本体は，発言の管理やアイデアの提示機能など，発想支援につながる機能を中心に管理し，具体的な作業は専用のアプリケーションに分担することで，ユーザに対しより効果的な作業支援空間を提供することが可能となる．

2.3 作業単位での再利用性

従来のグループウェアにおいて作業支援の中心は，協調作業の効率的な「収束性」にあった．この目的から KJ 法ツール[12]，状態遷移モデル[15]など発言の収束を促す機能を中心とするアプリケーションが多く開発された．しかし，従来のグループウェアでは，作業者が参加できなかった場合，作業結果を伝えることのみで，その結果にいたる「話の流れ」は示されない．また作業結果は,「話の流れ」の中の重要な情報が欠落して提供される可能性がある．本論文では，このような作業結果に対し作業間の連続性を保持する必要があると考えた．そこで，以下のように「イベント」,「作業過程」を定義する．

イベント 作業支援アプリケーションにおいて提供される機能を，ユーザが利用した作業に対し，作業単体としての再利用性と分散環境におけるコミュニケーション支援機能が付加されたもの

作業過程 イベントの生成順序と，イベントが生成されたアプリケーション情報が付加されたイベントの集合体

この定義によって，イベントは独立して管理可能となり，複数アプリケーションを利用する協調作業においては，アプリケーションごとのイベント管理，作業全体のイベント管理など，様々なイベントの組合せによる管理が可能である．また，イベントとアプリケーションの組合せによって，協調作業以外の作業へ再利用も可能である．

図2は作業過程の再利用を示している．DrawEditor は図を描くために，グループウェアで利用された子アプリケーションのひとつである．DrawEditor では「円を描く」,「四角形を作る」など作業が，イベントとして EventCollection へ保存されていく．作業終了時には，これらのイベントが集合体として作業過程を形成し，ある図形（作業結果）を示してしている．この EventCollection に保存されたイベントを PresentationTool に応用し，それぞれのイベントを順に評価を行うことで，DrawEditor での結果の形成過程を表現することが可能となる．再現可能なイベントによって，作業者は1つの作業から必要な部分を切り出し，異なる作業へ利用する作業融合が実現される．

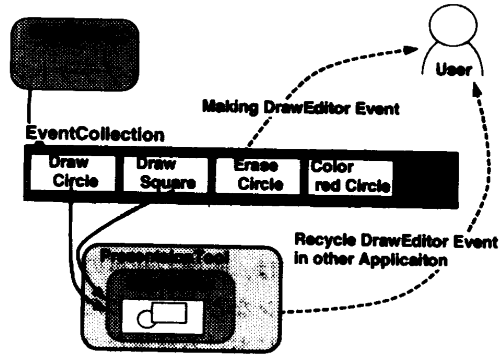


図2 発言の流れに含まれる発想支援情報と再利用性
Fig. 2 Idea supporting and reusablity in stream of statements.

3. FusionWorks

前章で述べたアプリケーションの応用機能，作業イベントの再利用機能を実現するために，コミュニケーション支援，具体的には通信システムの簡略化，また，再利用性を向上させるためにデータの一貫性を保証することが必要である．

以上の問題を解決するために，分散オブジェクト環境[1),5)~8)]において作業支援プラットフォーム FusionWorks[14)]を構築した．本章では，FusionWorks の構成とコミュニケーション支援機能，アプリケーション管理について述べる．

3.1 プラットフォーム構成

FusionWorks は，Smalltalk[4),17)]の仮想計算機空間上に構成される．Smalltalk を用いたのは，分散オブジェクト環境を実現するうえで重要な，クラスライブラリの統一を行う必要があることによる．これによって，FusionWorks 内で利用するデータすべてをオブジェクトとして統一して管理でき，データの一貫性が保証される．この FuisonWorks の全体構成を 図3 に示す．

FusionWorks は下位層より，ネットワーク層，OS 層，Smalltalk 層の3つに分類される．それぞれの層は下位層によって，各層におかれている各種機能モジュールに必要とされる機能補助を行っている．

FusionWorks の本体は Smalltalk 層によって実現される．内部の各機能モジュールは，メディアコントロールマネージャ（MediaControlManager: MCM）を中心にメッセージの配送管理を受け，必要とされるメッセージの送受信を実現している．MCM を介することによってメッセージの送受信形式は一般化され，個人作業空間，メディア提供空間，協調作業空間に生成されるアプリケーションに透過なアクセスが実現さ

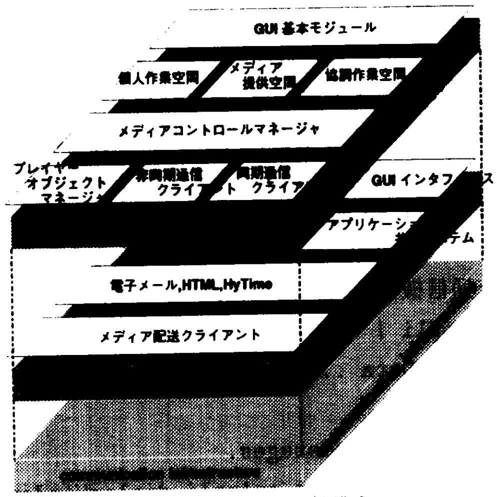


図 3 Fusion の全体構成
Fig. 3 Overview of "FusionWorks" system.

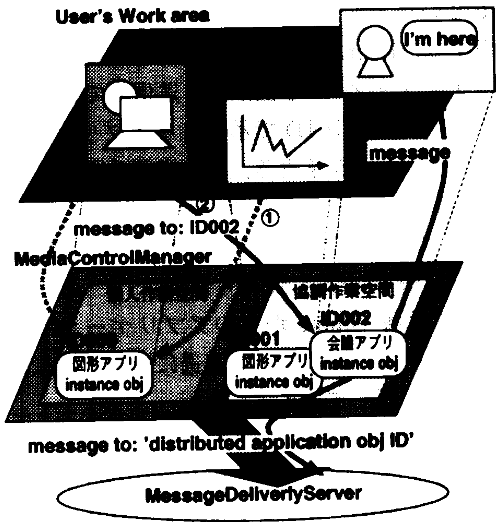


図 4 MCM におけるメッセージ配送管理
Fig. 4 Management of message delivery in MCM.

れる．MCM はメッセージの送受信要求に従って，同期/非同期通信クライアントを介し，メッセージの送受信を行う．ここで，メディア提供空間ではデータの再利用性を向上させるために，個人，協調の両作業空間で生成される作業情報を一括して管理を行い，ユーザに対して柔軟な情報提供機能の管理を行っている．

3.2 コミュニケーション支援機能

個人作業空間，協調作業空間で起動されるアプリケーション間のコミュニケーション支援機能の一般化には，メッセージ配送システムの簡略化を行うことが必要である[2),3)]．FusionWorks ではメッセージの配送システムを，アプリケーションに付加したアプリケーション ID によって，個人，協調の属性付けを行い，ID 管理によってメッセージ配送方法の一般化を行っている（図 **4**）．

メッセージは，「作業内容」，「アプリケーション ID」，「送信先リスト」の 3 つによって構成される．メッセージの具体的な役割は作業内容に包含され簡略化が実現される．図 4 において，①は個人作業空間のアプリケーションに対して，②は協調作業空間のアプリケーションに対してのメッセージの配送の流れを示している．

アプリケーションのメッセージ送信は，MCM にその依頼を行うことで実現される．MCM はメッセージを受信すると，メッセージのアプリケーション ID によって MCM 内で登録されているアプリケーション群の中からアプリケーションの実体を取り出す．送信されてきたメッセージは，そのアプリケーションに対して評価を依頼することで，作業内容が実行され配送作業が完了する．このときアプリケーション ID の属性が個人作業空間の場合はシステム内部で評価が完結

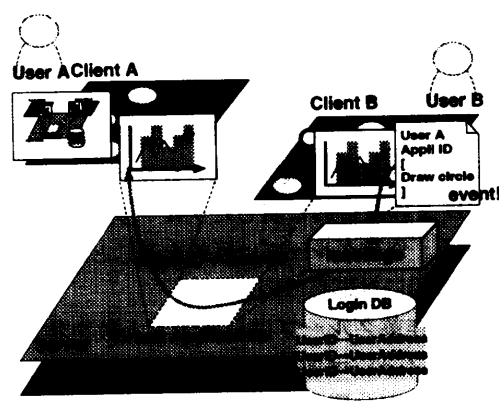


図 5 MessageDeliveryServer でのメッセージ配送管理
Fig. 5 Management of message delivery in MDS.

するが，協調作業空間の属性である場合は，送信されてきたメッセージは協調作業を行っているすべてのアプリケーションに対して送信を行う．メッセージの送信先リストは，メッセージの配送を行うために必要となる値である．協調作業空間へのメッセージは MCM によって MessageDeliveryServer（MDS）に送信される．MDS は送られたメッセージより，だれに配送するかを示している送信先リストに従って配送を行う．MCM がローカル環境内でのメッセージ配送処理を行うのに対し，MDS は分散環境での作業空間を管理し，コネクションレスなメッセージ配送処理を実現している．MDS からの処理の流れを 図 **5** に示す．

メッセージの流れは，協調作業空間に存在する図形ツールにおいて，UserB がツール上で「円を描く」という操作を行ったとき，その操作が UserA に伝えられる状況を示している．UserB からは「円を描く」という作業内容がイベント式で，「UserA」の「図形ツール」

へ送るというメッセージが生成される．MDS はメッセージより，UserA が登録している「図形ツール」のアプリケーション ID を検索し，送信先である UserA のアプリケーション ID で示されるアプリケーションに対してメッセージを送る．その流れによって UserA の作業空間にメッセージが渡され，ClientA の MCM 内で 図 4 の①と同様の処理が行われ，UserB からのメッセージが UserA に伝えられる．

MDS では各 Client からのアプリケーションへのメッセージ送信要求に対して透過に行うために以下に示す情報を管理している．

ユーザのログイン情報

作業システムに対して，現在だれがログインしているかを管理している．ログイン情報として，ログインホスト，使用ポート，電子メールアドレス情報，ユーザ名を管理している．ユーザに対してはこのような情報を隠蔽し，ユーザ名のみで指定させる．

作業グループ情報

作業グループはユーザの作業空間を管理する各 Client において，複数起動されたアプリケーション ID を作業目的に属したグループ ID として管理する．作業グループは目的を同じとする少数の作業集団に対応している．

ここで，作業グループは，属する人数を特に決定せずにオープンに参加可能なものと，あるメンバにのみその存在が公開される隠蔽情報によって参加可能なものの 2 種類が存在する．前者をオープングループ（OpenGroup），後者をクローズドグループ（ClosedGroup）と定義する．

これらの情報を各 Client に必要に応じて提供することで，ユーザ，アプリケーションに対しネットワーク透過なコミュニケーションが実現される．

3.3 イベント再利用性を考慮した通信プロトコル

前節で述べたコミュニケーション機能の簡略化と作業単位での再利用性を向上させるために，通信プロトコルにおいてこれら 2 つの条件を保証する必要がある．コミュニケーション機能の簡略化については，前述した「作業内容」,「アプリケーション ID」,「送信先リスト」のメッセージ形式で実現され，以下に示すような記述となる．

```
MediaControlManager
    sendEvent: 作業内容
    appliName: アプリケーション ID
    to: 送信先リスト
```

メッセージはアプリケーション ID の属性によってその配送先が決定されるため，ローカル環境，分散環境を一般化しての記述が可能である．また，送信先リストは全員に送信する場合は，# (#all) と記述し，また具体的に指定する場合はそのユーザ名（# ('木本陽介')）を明示的に記述することで送信が実現される．

作業内容は 2.3 節でのイベントが記述されている部分である．イベントが再利用可能な形式で生成され，柔軟な通信機能を持つために，

```
[:anAppli |
 | inputData ... |
  ...

    anAppli messages.
    inputData := anAppli dialogWindows.
    inputData fotNextComs.
  ...

    ^returnValue
]
```

に示すように Smalltalk でのブロック式で記述する．Smalltalk の仮想計算機環境によって，ブロック式はイベントとしての再利用性，汎用性を持つオブジェクトとして保持が可能となる．

ブロック式内部の記述は代入アプリケーション（**anAppli**）へのメッセージ集合となっている．プログラム開発においてはターゲットとするアプリケーションに，メッセージ（**messages**）によって必要とするオブジェクト（**inputData**）を獲得し，それに対して新たなメッセージ（**forNextComs**）を送ることで，ユーザとのインタラクティブなイベントなど，複雑な作業が記述可能となる．代入アプリケーションは分散環境上のユーザ作業空間において得られ，通信中はアプリケーション非依存であるため，イベントとしての柔軟な通信機能が確立される．またブロック式は，メッセージ **value**, **value:** が送信されるまでその評価を遅延する機能を持ち，代入アプリケーションが指定されるごとにイベントが再現され，イベントとして再現が可能となる[9),10)]．イベント（ブロック式内部）の記述にはオブジェクトの性質にもよるが，基本的にはウインドウの生成や作業支援機能の制御，また，作業者が生成した図形オブジェクトなど様々な記述，オブジェクトの包含，また必要に応じてその結果を返す（**returnValue**）ことが可能である．ブロック式はアプリケーション ID で指定されるアプリケーションオブジェクトを獲得し，評価されることで作業が実行される．

以上の定義に従い，具体的にプログラムを行った例を **図 6** に示す．

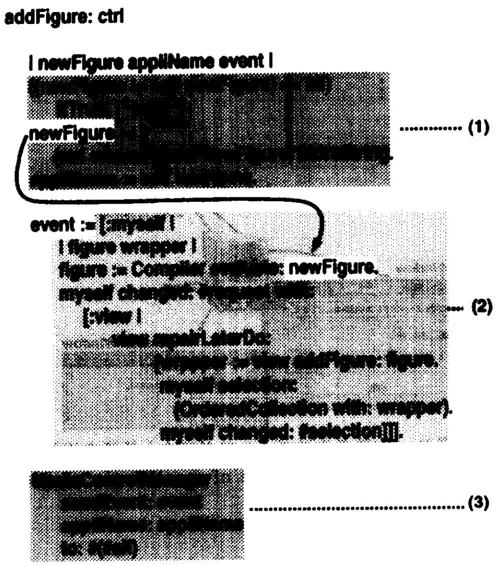


図6 メッセージ内容の例
Fig. 6 An example of contents of message.

(2) は,「描画する」という作業内容が *appliName* で指定されたアプリケーションへの遅延処理としてイベント化されている. ブロック式内部の ***myself*** が, MCM においての評価時にアプリケーションオブジェクトが代入される変数である. 作業内容自身は Smalltalk のメソッドの一部として定義され, メソッド内で利用可能なオブジェクトをすべて作業内容に包含することが可能である. 実際にイベントとして送信, 保存されるのはこの (2) である. (1) は *newFigure* を (2) のメッセージ内への代入前処理を行っている. この代入前処理はオブジェクトのシステムへの依存性を省くための処理である. 依存性を持つオブジェクトとは, システムが利用しているウインドウシステム, OS に強く依存し, その空間より移動不可能なオブジェクトである. (3) は (2) で定義されたイベントを MCM に対し送信処理を依頼している部分である.

図 6 の記述形式によって自由な作業内容の記述と, 作業の変化にともなって生成されるオブジェクトを同時に保存可能としている.

3.4 アプリケーションの作業空間と依存関係管理

アプリケーションは MCM において, 個人作業空間, 協調作業空間の分類のために 3.2 節で述べた作業グループ情報を, 協調作業空間に起動されるアプリケーションに付加することによって管理される. また MCM はこのようなアプリケーションに対し, 2 章で述べた依存関係の管理を行う. 依存関係によってアプリケーションは親アプリケーション, 子アプリケーションに分類される.

MCM に対してアプリケーションの生成依頼をすると, 親アプリケーションとしての性質を持ってアプリケーション ID が付加され生成管理される. アプリケーションは作業空間における支援機能を定義したクラスより生成されるインスタンスオブジェクトである. 親アプリケーションの管理はアプリケーション ID にその実体となるアプリケーションオブジェクトと子アプリケーションリストで構成される. また, 親アプリケーションが協調作業空間の属性を持つ場合は, 子アプリケーションにも属性が反映される.

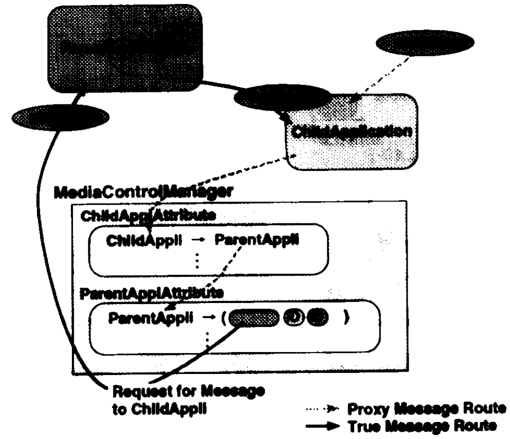


図7 アプリケーション依存関係によるメッセージ配送
Fig. 7 Message delivery by dependency in applications.

子アプリケーションの属性は MCM に対し生成依頼を行うが, アプリケーションオブジェクトの管理を親アプリケーションで行い, 親アプリケーションの属性は子アプリケーションリストにアプリケーション ID を登録することで管理され, 依存関係が確立される.

図 7 に以上の機能によるメッセージ配送の流れを示す. メッセージの生成方法はアプリケーションに依存するが, 図 7 においては子アプリケーションからのメッセージが親アプリケーションに渡され, 親アプリケーションのメッセージに包含されて送信されている. このように, アプリケーション間の親, 子の依存関係が確立される.

4. FusionWorks 実装

前章までの構成を基に, SparcStation 上で VisualWorks R1.0 を利用して学内 LAN 上での FusionWorks の実装を行った. 本章では, FusionWorks の通信システムおよび作業グループ管理について述べる.

4.1 オフィス業務に対応したネットワーク構成

本論文で提案する FusionWorks は, 具体的な業務を実行する作業グループを定義し, 作業者が目的を同じとする作業グループに参加することで作業を行う. このグループ内において各作業者が 1 台端末を所有し,

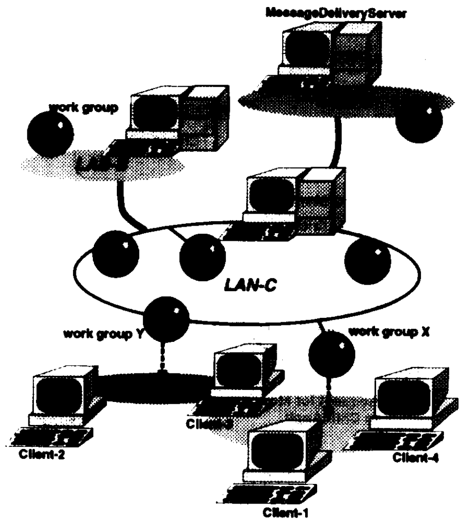


図8　FusionWorks 実装環境
Fig. 8　Environment of FusionWorks implementation.

その端末がLANで結ばれている．

図8のLAN-A～Cは企業内の部署を，球が作業グループを，Client-1～4はFusionWorksによる作業空間を示している．LAN内にはMDSを1つ起動し，作業者のログイン管理，作業グループ管理，メッセージ配送管理を行う．図8において，Client-2，3が作業グループXに，Client-1，3，4が作業グループYに参加している．ここで，Client-3が2つの作業グループに同時に参加しているが，これはオフィス業務が必ずしも1つとは限らず，複数の業務を担当する場合に対応するためである．

任意の作業グループへの参加，また各作業者の作業空間管理を行うために，作業者のログイン管理を各Client，MDSにおいて行っている（**図9**）．

作業者からのログイン要求は，作業空間上のログインウインドウで入力された情報を基に使用ホスト名を付加して，`loginOS:`でClientよりMDSに伝えられる．MDSはその情報に，使用可能な通信情報を発行してログインデータベースへ格納し，すでにログインしたユーザ（既ログインユーザと呼ぶ）の情報を付加して`loginAcceptOS:`でログイン通知メッセージがClientに伝えられる．このメッセージを受信したClientは，メッセージに記述されている通信情報より非同期メッセージ受信プロセスを起動する．また，MDSは既ログインユーザの各Clientに対し，新規にログインしてきたユーザ情報を`loginNewUserAcceptOS:`で送信する．以上の処理によってユーザのログイン処理が完了し，各Client間でのコミュニケーションが保証される．

同様にログアウトは，`logoutOS:`がMDSに送信さ

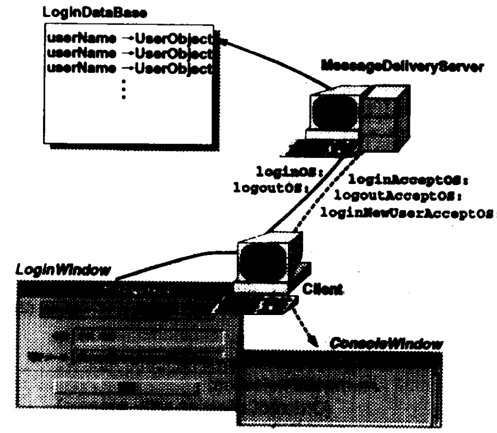


図9　作業者のログインログアウト管理の流れ
Fig. 9　Flow of management of user's login and logout.

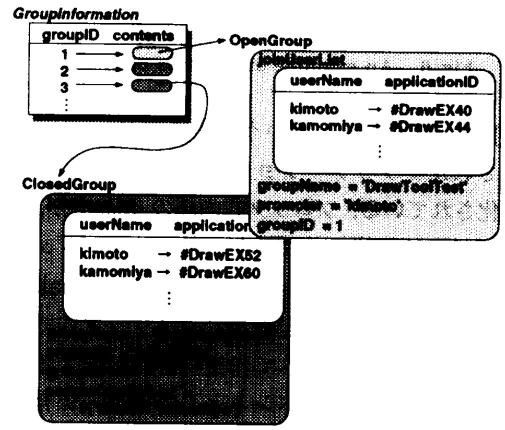


図10　作業グループ管理
Fig. 10　Management of work group.

れるとログインデータベースからユーザ情報の削除が行われる．また同時に，ログイン中のユーザにログアウトユーザを知らせるために，`logoutAcceptOS:`で各Clientのログインデータベースの更新を行う．

4.2　作業グループ管理

一般にオフィス業務では，作業目的別に3～7人程度の少数グループによる協調作業が複数構成され作業が行われる．作業グループはその少数グループを分散環境において実現するためのものである．

図8の作業グループX，Yは，LAN-C内のMDSで管理空間が生成されている．**図10**はユーザの作業グループへの参加が行われたMDSでの作業グループ情報を示している．

作業グループは協調作業アプリケーションの生成時にその登録が行われ，登録形式は2種類存在する．1つは属する作業グループが存在する場合である．この場合は，その作業グループを指定して参加をするので，指定した作業グループへの参加情報を送信することで

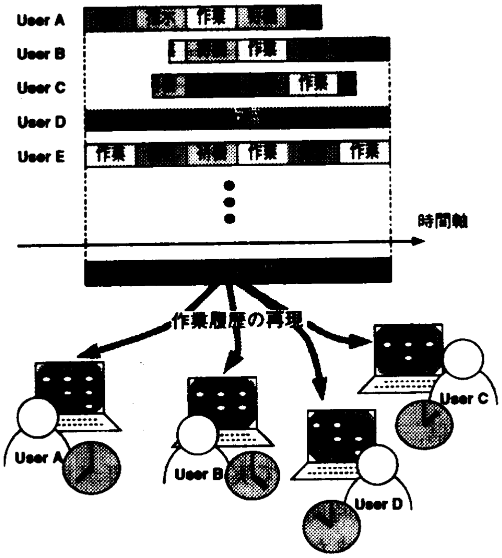


図 11 協調作業の非同期性
Fig. 11 Asynchronous cooperation.

処理が完了する．もう1つは属する作業グループが存在しない場合である．この場合は，新規に作業グループを登録する必要があるため，参加情報に「新規作業グループ情報」が付加される．新規作業情報は，作業グループを生成する際に必要とされる作業グループ名，発起人，メンバーリストである．作業グループ名，発起人はそれぞれ目的，グループ立案者を示す．メンバリストはクローズされた特定のグループを生成するときに，参加認証に必要とされる情報である．

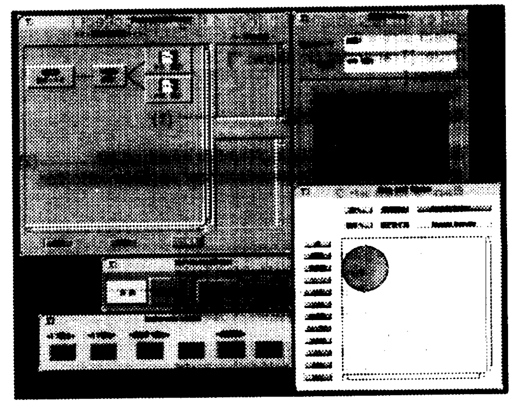
作業グループへの参加情報は作業グループ ID，アプリケーション ID，ユーザ情報が MDS に送信されることで，図 10 のオープングループ，クローズドグループの形でそれぞれの情報が関連付けられる．協調作業空間におけるメッセージ送信処理は作業グループ ID によって管理されている．協調作業アプリケーションは生成時に作業グループ ID の属性を持つことで，その作業グループに属するユーザの各 Client へメッセージ送信が可能となる．また作業グループ管理によって，ユーザの都合による不在，途中の入退場に対して，メッセージ送信履歴の管理が作業グループ単位で可能となる．

5. 作業融合による会議支援システムへの応用

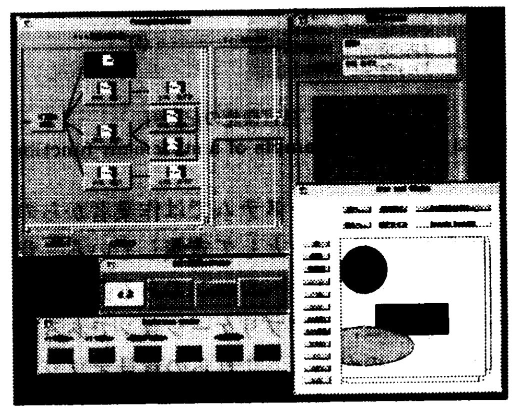
FusionWorks の作業再利用機能，また作業融合機能の適応性を確認するために，会議支援システム[16]への実装を行った．本章ではこれらの機能の有効性について述べる．

5.1 時間と作業の共有

グループウェアに関する研究で様々なシステムが提案されてきた．しかし，従来の会議支援システムでは，作業共有を時間共有を前提としていたこと，また作業履歴が会議経過を単純に記録した点が作業支援システムとしての能力を限定していた．これは，メンバ全員を特定の時間に拘束することが困難であることや，作業履歴の再利用を十分に考慮しなかったためと考えられる．これらの問題を解決するために，FusionWorks の作業再利用機能を利用する．



(a) 作業イベントの評価時



(b) 作業イベントの評価時

図 12 作業の再現と再利用
Fig. 12 Recycle and reuseability of workevent.

図 11 は，作業参加者の協調作業が行われた過程を横軸にとり，その過程で生成された履歴を User A～D がそれぞれのスケージュールにおいて作業への参加を示している．作業再利用性機能を利用することで，実際の協調作業に参加をしていない場合においても過去の時間と作業が再現され，作業全体の流れが把握可能となる．またアプリケーション依存関係によって，新たな作業に具体的な再利用を行うことが可能となる．

図 12 は本論文で実装を行った会議支援システムにおいて，図 12 (a) から 図 12 (b) において，意見ノードが生成されると同時に，図形エディタにおいて具体的に図形が形成される過程が再現されていることを

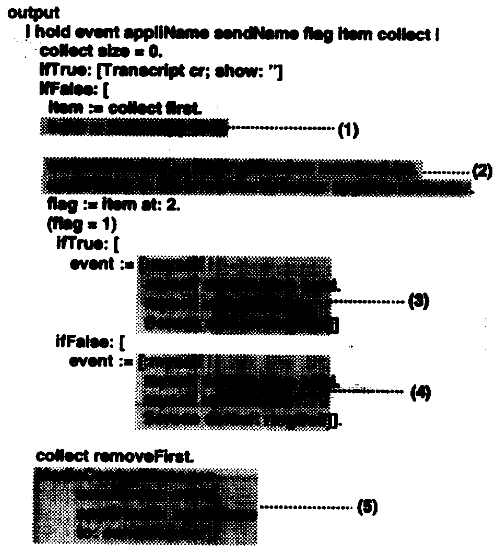


図13 発言機能の記述例
Fig. 13 A script example of a statement function.

示している．会議支援システムでは作業者からの各意見を，意見オブジェクトとして定義している．意見オブジェクトは親アプリケーションである会議支援システムからのものと，子アプリケーションである図形エディタからのものがある．これらの意見オブジェクトが意見ノードとして会議支援システムで管理される．意見ノードは依存関係と作業イベントとして再利用性によって，会議終了後に会議の流れを再現することが実現される．また，子アプリケーションからの意見として別管理を行うことで，子アプリケーション専用のイベントとして会議に依存しない作業としての再利用も可能となる．

5.2 柔軟な作業支援機能の利用

図12に示される会議支援システムには，FusionWorksによる作業イベント機能が十分に応用されている．**図13**は，会議システムにおいて発言機能を支援する部分である．図6 (2) と 図13 (3)，(4) を比較することで，ブロック式内部がまったく異なるのに対し，コミュニケーション機能が柔軟に実現されることが理解できる．FusionWorksによって，コミュニケーション機能の細分化を行わなくとも複雑な処理が実現される．また，各作業支援機能において条件に応じた記述が可能であり，コミュニケーション機能の簡略化が実現されている．

5.3 実装評価

オブジェクト指向環境における協調作業のための試作アプリケーションの実装において，発言機能と協調作業機能をそれぞれ専門特化した従来システム[15]と，協調作業機能とアプリケーションを分離しそれぞれの汎用化を行った新システム (FusionWorks) での比較評価を**表1**に示す．発言機能は，発言ウインドウ，発言文字，転送，表示の4つの機能を管理し，それを4人のユーザが利用する環境を想定している．

表1 発言クラス実装評価
Table 1 An evaluation of statement class impremetation.

	通信クラス	通信機能依存度	開発時間
従来システム	1	4	4
新システム	1	1	1

表1は各項目を従来システム，新システムにおいて比率で示している．従来システムでは，協調作業空間専用の通信，発言クラスを利用したため，発言ウインドウ，転送機能を独立して実現する必要があった．これによって，開発時間も比例して増加し，システムが複雑化している．また，協調作業機能に依存したため，イベントの再利用性も実現不可能であった．

これに対し新システムでは，通信機能，アプリケーションの依存度を減らし，それぞれを独立しコミュニケーション機能の提供という形式をとることで，依存度に比例し開発時間が短縮された．これは，プログラム開発において通信機能について深く考慮する必要がなく，図6のようにイベント記述部，通信部と明確な分離による記述によって，自由な開発環境が実現できたためと考えられる．以上のような傾向は試作システム全体に現れており，試作アプリケーションの通信システム全体の実装においては，従来システム (約3人 × 一カ月)，新システム (約3人 × 1週間) であった．

6. まとめ

本論文では従来までのグループウェアの問題点を抽出し，柔軟な拡張性，作業融合環境を実現するFusionWorksについて述べた．その特徴を以下にまとめる．

- コミュニケーション機能と作業支援機能の分離
- 柔軟な拡張性と再利用性を持つ通信プロトコルの確立
- 作業グループによる作業空間の管理

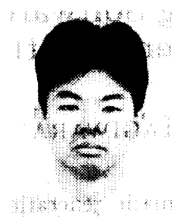
今後の課題として複数協調作業への対応を考えている．一般社会の仕事では複数の協調作業への参加は，作業者が物理的に1人しか存在しないため難しい問題である．しかし，協調作業の程度にもよるが，作業の流れを監視しているだけの場合もあれば，積極的に作業に参加する場合もある．このような性格の異なる作業においては，同時に参加を行った方が作業効率が向上すると考えられる．複数協調作業への参加を実現するために，コンピュータ上での仮想作業空間において

複数作業空間の管理を行うことで，物理的に1つの存在である作業者を複数の作業において存在させることが可能となる．またこれによって，協調作業空間間での情報の再利用が実現され知的触発も期待できると考えられる．

参考文献

1) 乾　和志，菅原圭資：分散 OS Mach がわかる本，p.223, 日刊工業新聞社 (1990).
2) 日本サン・マイクロシステムズ株式会社技術サポート部，マーケティング技術部，製品企画部：Java 言語環境 A White Paper，p.60, 日本サン・マイクロシステムズ (1995).
3) Sun Microsystems inc: HotJava Home Page, <http://java.sun.com>.
4) 上谷晃弘：統合化プログラミング環境—Smalltalk-80 と Interlisp-D—，p.354, 丸善 (1987).
5) Open Software Foundation: OSF DCE 分散コンピューティング環境，アジソン・ウェスリイ・パブリッシャーズ・ジャパン (1993).
6) 湯浅　敬，プラーティブ．K. シンハ：ネットワークマルチメディアアプリケーションのためのオブジェクトモデル，日本ソフトウェア科学会 WOOC'92，pp.347–359 (1993).
7) 米沢明憲，松岡　聡，加藤和彦 (編)：オブジェクト指向コンピューティング II WOOC'93，p.254, 近代科学社 (1994).
8) 尾内理紀夫：オブジェクト指向コンピューティング III WOOC'95，p.171, 近代科学社 (1995).
9) ParcPlaceSystems Inc: ObjectWorks Smalltalk User's Guide, p.434, ParcPlaceSystems Inc. (1992).
10) ParcPlaceSystems Inc: VisualWorks クックブック，p.701, ParcPlaceSystems Inc. (1995).
11) 松下　温，岡田謙一，勝山恒男，西村　孝，山上俊彦 (編)：知的触発に向かう情報社会—グループウェア維新—，p.317, 共立出版 (1994).
12) 中島一彰，早川栄一，並木美太郎，高橋延匡：「紙」メタファによる手書きコミュニケーションと分散手書き KJ 法システム，情報処理学会，DPS 61-22 (1993).
13) 川村渇真：オブジェクト指向コンピュータを作る，アスキー (1993).
14) 木本陽介，鴨宮　淳，川畑　豊，服部進実：オブジェクト指向ヒューマンコンピューティングによる知識ナビゲーションの一考察，情報処理学会，DPS 68-9 (1995).
15) 木本陽介，新井敏正，服部進実：状態遷移会議モデルによるマルチメディア分散協調作業支援システム，信学技法，HC 93-87 (1994).
16) 木本陽介，服部進実：離散時間共有型協調作業支援システム，情処学会マルチメディアと分散処理ワークショップ，W-DPS95 (1995).
17) GoldBerg, A. and Robinson, D. (著)，及川一成ほか (訳)：Smalltalk-80—言語詳解—，p.569, オーム社 (1987).
18) 青山幹雄：コンポーネントウェア：部品組み立て型ソフトウェア開発技術，情報処理，Vol.37, No.1, pp.71–79 (1996).

(平成8年3月27日受付)
(平成8年9月12日採録)



木本　陽介（正会員）

1972年生．1994年金沢工業大学工学部情報工学科卒業．1996年同大学工学部情報工学専攻修士課程修了．同年ソニー（株）入社．現在，分散オブジェクト指向システムの開発に従事．日本ソフトウェア科学会会員．

服部　進実（正会員）

1964年東北大学工学部電子工学科卒業．同年より富士通（株）にて情報通信システムの研究開発部門に所属．1989年金沢工業大学情報工学科教授．1994年同大学人間・情報・経営工学系長．工学博士．情報通信システム，分散処理システムの研究に従事．電子情報通信学会，人工知能学会，IEEE 各会員．