

## 同期型グループウェア開発環境の開発と適用

太田哲生 平山愛子 貫井春美  
(株)東芝 研究開発センター  
システム・ソフトウェア生産技術研究所

筆者らは、同期型グループウェア (GW) の代表である共有ウィンドウをベースとするデスクトップ会議システムを試作し試行した。試行から得た課題を基に同期型 GW の開発を容易にするために開発環境として ReGAPI(Real-time Groupware Application Program Interface) を構築し、その上で新たにデスクトップ電子会議システム ( $D^2 - TCS(ver2)$ ) を開発した。ReGAPI を用いることにより複数の異なったプラットフォームへの対応や高い応答性を備えたシステムの構築が容易になる。本稿では、ReGAPI の設計思想、同期型 GW のアーキテクチャ、および ReGAPI の適用事例としての共有ウィンドウシステムの実現と評価について述べる。

## Design and implementation an application program interface for real-time groupware

Tetsuo OHTA, Aiko HIRAYAMA, Harumi NUKUI  
Systems & Software engineering, Laboratories.  
R & D Center, Toshiba Corp.  
70 Yanagi-cho, Saiwai-ku, Kawasaki 210, JAPAN

We are studying and developing practical groupware to support software development at present and in near future. We have designed and developed an application program interface for real-time( or synchronous) groupware which working in heterogeneous multi-platforms , we named which is ReGAPI(Real-time Groupware Application Program Interface). We use a shard window system which are developed using the ReGAPI, with Audio/Image communication systems in actual distributed software development. In this report we describe a design concept and architecture of ReGAPI, an implementation of sharing applications with our middle-model, a shard window system which is application of the API and evaluations.

## 1 はじめに

筆者らは、複数の拠点に分散する環境下でのソフトウェア開発を支援するための分散型開発環境  $D^2$ <sup>\*</sup>[1] の研究を進めており、コミュニケーションを非同期 [2]、同期 [3] の両面から支援するシステムの開発および適用を行なっている。同期型システムは、国内数拠点で試行し有効性の確認と課題を抽出した。本稿では、試行による課題を基に新たに設計・開発したグループウェア開発環境<sup>†</sup>と、それを用いて構築したデスクトップ電子会議システム<sup>‡</sup>、について報告する。

## 2 設計思想

筆者らは、共有ホワイトボードをベースとした同期型コミュニケーション支援システム<sup>§</sup>[3]を開発してきた。同システムの国内数拠点(東京、仙台等)での試行結果から、応答性の向上、ファイルおよび他プロセスとのデータ交換の実現といった課題が抽出された。これらは、現状のネットワークの性能やシステム自身のアーキテクチャに起因する。これらの課題を解決し、現在および近い将来の分散ソフトウェア開発において実用的かつ有効な同期型コミュニケーション支援を実現する必要がある。

### 1. 既存ツールとのインタフェース

これまでユーザが利用して来た既存の個人向けアプリケーションプログラムを会議の場で利用できると、わざわざグループウェア用にデータを作成する必要がなく、そのグループウェアの受容度が高くなる。

### 2. 低速な WAN における実用性

高速化する LAN に比べ、WAN は現在もなお低速である。この低速な WAN 上で実用的な通信量と応答性が必要である。

### 3. 複数のプラットフォーム

ソフトウェア開発全般では多様なプラットフォームが利用される。例えば、工程により支援ツールや利用環境が異なる。また、主にセルフ開発が中心となるため対象製品(部門)による開発環境の相違が存在する。データ形式の相違を吸収するために、資料

<sup>\*</sup> $D^2$ : Distribute environment for software Distributed development

<sup>†</sup>Real-time Groupware Application Program Interface

<sup>‡</sup> $D^2$ -TCS:  $D^2$  - desktop Tele-Conference System(ver2)

<sup>§</sup> $D^2$  - TCS(ver1) 等

は紙に出力し配布されている。このため、プラットフォームに依存しない支援が重要となる。

### 4. 記録の重要性

従来の同期型コミュニケーション支援システムでは、リアルタイムに情報を共有することに特に重点を置いていた。一方、ソフトウェア開発の主な作業は非同期で行なわれる。このため非同期型とのインタフェースとして、データ(対話内容)の記録機能が重要となる。加えて、記録を残すことは、電子化や再利用性への道でもある。

## 3 アーキテクチャ

### 3.1 アプリケーション共有

複数のユーザが共同作業を行なうアプリケーション(AP)には、次の2種類が考えられる。

- シングルアプリケーション(以下 SAP)の共有  
本来個人使用を目的に設計されているアプリケーションを、複数人で利用できるようにする。
- グループウェアアプリケーション(以下 GAP)  
あらかじめ複数のユーザで操作されることを前提に設計/開発されたアプリケーション

我々は、主に後者の共有アプリケーションの開発を目指す。

### 3.2 AP 共有の実現方式

AP 共有の実現方式は XTV[4]、OGM[5]、COGNET-SHARE[6] などのように実行するアプリケーションが1つであり、出力(イベント)を分配する集中型と、MERMAID[7]、ASSOCIA[8] などのように複数の(複製)アプリケーションが実行し、入力(イベント)を分配する分散型がある。前者は、状態の一貫性を管理するのが容易な反面、性能および表示画面の自由度や他プロセスとの情報交換に問題がある。後者は、逆に性能面および自由度の面は優れるが、同期制御が難しく、複製の実行環境の統一、および途中参加などが問題となる。

我々は、アプリケーション実行は複製型同様に複製を用いて行なうが、分配するデータを入力イベントでなく、抽象度の高い独自独自プロトコル(中間プロトコル)を用いる中間型を提案する。中間プロトコルは、特定の入力イベントのみに対応し、他の複製に対して必要な場合のみ同報を行なう。各 AP には、中間プロトコルを処理

するMM(Message Manager)を組み込む。すなわちアプリケーション変更を方式する方式である。

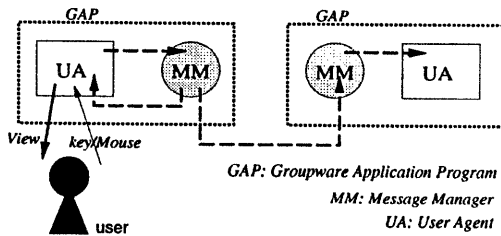


図 1: 中間型によるアプリケーション共有

### 3.2.1 メッセージ集中化方式による一貫性制御

メッセージの同報は、集中型同様に単一のキューにメッセージを集め順序付けすることにより、一貫性の制御を行なっている。この場合、手で発生したメッセージは、キューとの間を往復した後に処理されるため、応答性の低下がみられる。特に WAN 環境で利用する場合、メッセージの伝送遅延が大きく影響し、対話処理に不向きとなりやすい。

### 3.2.2 ローカル先行処理による応答性向上

応答性低下問題を解決するために“ローカル先行処理”を採用している。先に述べたように入力イベントは、他の複製に伝える必要のあるイベント (global event) と独自に処理するイベント (local event) に分類され、中間プロトコルは global event に対応させることで、通信総量の削減をしているが、加えて図 2 に示すように、中間プロトコルのメッセージ処理を非同期的に行なうことで、応答性の改善を行なっている。この方式を採用した場合、メッセージの到着順序の一貫性に問題を生じる恐れがある。しかし、実運用では、ほとんどの場合に問題は生じなかった。これは、以下の理由によるものと思われる。

1. イベントの発生源が人間であり、発生速度が遅い。
2. メッセージの宛先を、各複製 AP ではなく各複製 AP 内の各オブジェクトとしており、原則的にオブジェクト毎のメッセージ順序の一貫性が保持されればよいために、衝突が発生しにくい。

ただし、まれに乱れが発生する恐れがある。このため、検出手段と明示的に操作権を処理する機能を設け対応し

ている。ただし、通常は操作権の取得を必要としない。

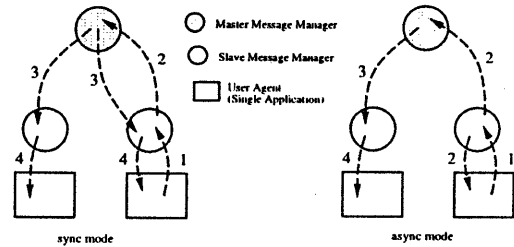


図 2: 中間プロトコルの同期/非同期処理

## 4 ReGAPI と $D^2 - TCS(ver2)$

中間型の共有方式を用いてグループウェアの作成を支援する API として ReGAPI (Real-time Groupware Application Program Interface) を設計/実装した。同 API を用いて、在席型の遠隔会議システムを実現し、現在ソフトウェア開発に適用している。

### 4.1 ReGAPI: Real-time Groupware Application Program Interface

ReGAPI は、以下の機能を提供する。

1. 接続/切断処理、参加/退席処理
2. メッセージの同報処理など
3. オブジェクト管理 (ID 管理、複製など)

ReGAPI の構成を、図 3 に示す。ReGAPI は、GPM: Groupware Application Program Manager、MM: Message Manager、PM: Port Manager により実現している。また、ReGAPI は、ウィンドウプログラミングのツールキットである Tcl/Tk<sup>¶</sup>[9, 10] 上に構成している。ReGAPI を利用するプログラムは、Tcl のスクリプト (一部拡張) で記述する。Tcl のコマンドから ReGAPI の機能が呼び出され処理を行なう。ReGAPI を利用した事例として、TCS の共有ホワイトボードである white board, negotiation board 等がある。

以下に、ReGAPI の中心となる GPM と MM について記述する。

<sup>¶</sup>Tool Command Language/Tool kit

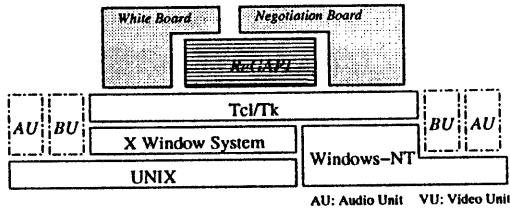


図 3: Real-time Groupware Application Program Interface の構成

#### 4.1.1 GPM:Groupware Application Program Manager

GPM は、GAP の起動等の管理を行なう。これにより、会議中に新たな GAP 等を起動した場合に、個々のアプリケーションは論理通信路の設定やグループ管理を行なう必要がない。

#### 4.1.2 MM:Message Manager

MM は、主にメッセージ転送に関する処理を行なう。

1. メッセージの同報処理など  
必要な複製へメッセージの転送を行なう。
2. オブジェクト (ファイルやプログラムなど) の転送  
オブジェクトは、グローバル ID とローカル ID で管理されており、その間の変換作業を行なう。また、外部プログラムの呼び出しでは、実行環境毎のコマンド名テーブルを持ち、その変換を行なう。通信量の削減をするため、オブジェクトの圧縮転送、およびキャッシュ機能を提供する。
3. データおよびプログラム形式の管理  
データを転送先で可読な形式に変更する。

#### 4.2 共有ホワイトボード

先に述べた ReGAPI を利用して、共有ホワイトボードシステム ( $D^2 - TCS(ver2)$ ) を開発した。図 4 に、開発した共有ホワイトボードの画面例を示す。

1. ジェスチャー機能  
マウスによる描画、キーボードからの文字入力、テレポインティングなどの awareness および annotation 機能を有している。ある。個々のコメントなどは、描画オブジェクトとして記録されており、再

配置、および再編集が可能である。また、複数のコメントを重ね合わせることも、可能である。システムには個々のコメント毎に作成者および、作成時刻などが記録されている。

#### 2. 操作権制御 (floor control)

どのメンバーも、常に自由に操作可能な方式を原則としている。異常時には主として音声コミュニケーションを利用して回復をはかっている。

#### 3. シングルアプリケーションの共有

TCS の共有ホワイトボードには、SAP の動的な共有機能を含めず、他の AP 画面のスナップショットを簡単に取り込む機能のみをを実装している。取り込んだイメージは、先に述べた描画オブジェクトとして扱われる。また、この一連の操作は、簡単なマウス操作とし、コミュニケーションの連続性を保持している。

#### 4. 外部とのデータ交換

ReGAPI の機能を利用することにより、テキストファイル/イメージファイルの人力を可能としている。各データの形式は、ReGAPI 内で各複製に適した形式に変換する。

## 5 評価

#### 1. ReGAPI

$D^2 - TCS(ver2)$  の開発に適用して ReGAPI を利用することで、共有機構の実現が容易に出来ることが確認された。今回作成した作成  $D^2 - TCS(ver2)$  が十分な性能を有することから、ReGAPI を用いたグループウェアの性能面の実用性を確認した。ReGAPI 上に作成したグループウェアは、ほとんど無修正で異なるプラットフォーム上で動作している。

#### 2. $D^2 - TCS(ver2)$

現在、音声/画像通信機能と併用し、遠隔地間での仕様レビューや進捗管理に利用している。開発中のアプリケーションの画面情報や、設計情報格納した CASE ツールのデータ表示のため、イメージ取り込み機能が良く使われている。各拠点は、64K ~ 1.5Mbps の専用線あるいは、国際 ISDN 回線により接続されている。性能面、特に応答性に関しては高い評価を得ている。また、会議内容が記録され資料として残る点の評価も高い。

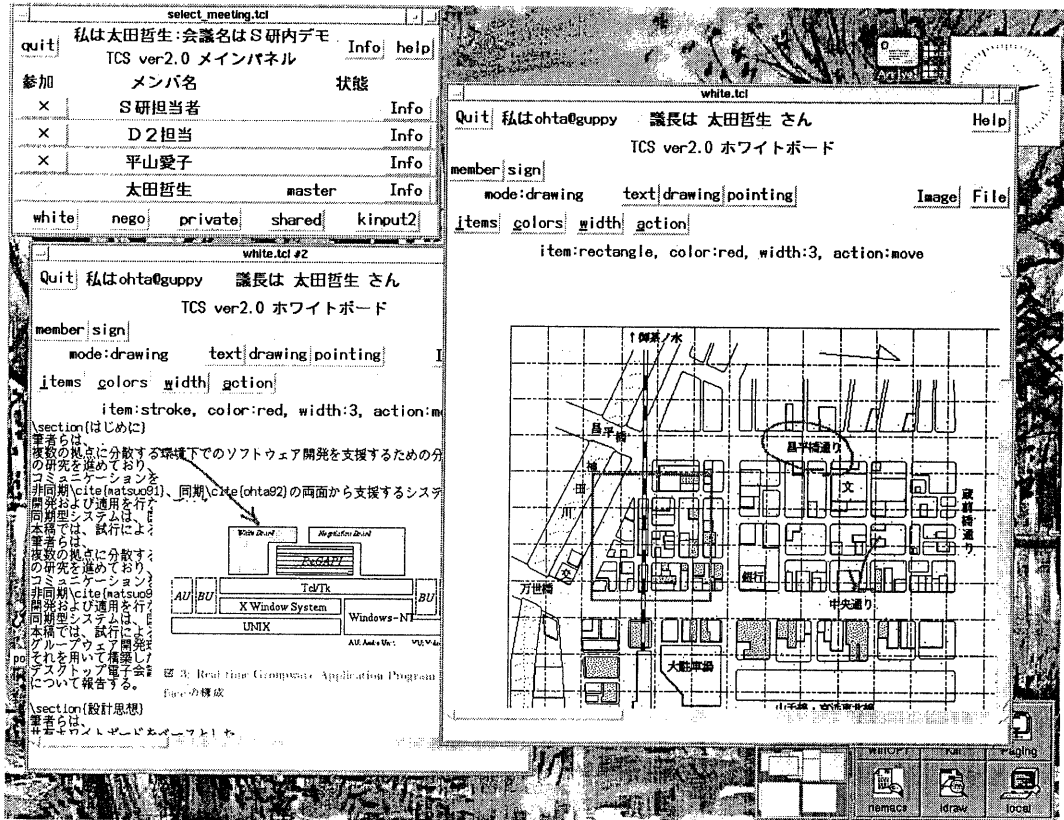


図 4:  $D^2 - TCS(ver2)$  の画面例

## 6 考察

### 1. アーキテクチャと性能

中間型アーキテクチャの採用により、通信データ総量の削減と高い応答性を実現できた。途中参加や参照オブジェクトの転送処理等を専用に行なう ReGAPI をグループウェアから切り出すことにより、個々の同期型グループウェアの開発を容易にした。

### 2. 既存アプリケーションの (AP) 共有

今回のシステムでは、既存の AP を動的に共有はしていない。SAP の共有により、メンバー間で AP の操作および動作のチェックが可能となることの利便性は、筆者らも認めるところである。ところが、実際に集中型の SAP 共有機構のテストを行なったが、この機能はあまり使われなかった。理由については以下の項目が考えられる。

- 現状の動作環境ではかなり負荷が高く、対象 AP の応答性が悪い。

- 実行中の AP に一時的にコメントを付けることは出来るが、記録として残すことが出来ない。
- AP の動作 (スクロールなど) により、コメントと AP の状態が不一致になる。

加えて、試行を通じた分析により、ユーザの要求には次の 2 つの目的が含まれている

- (a) SAP の動的変化を相手に伝える
- (b) SAP のデータを共有する (確認する)

従来グループウェア開発では (a) に着目していたが、実際のニーズは (b) が強いことが分かったからである。この点を考えれば、スナップショットを共有ウィンドウに取り込む view の共有で十分である。この方式では、リアルタイムに変化する AP に対しては、連続的にスナップショットを取り、これを並べ説明しなければならない。一見、非常に不便だと思われるが、スナップショット方式では、あらかじめ連続対象の希望部分が選択されている点で、S/N 比が非常に高く、必要記憶容量も格段に少なくて済

む。合わせて結果の一認性も高く、記録として有利である。しかし、共同デバッグなどでは、動作状況をリアルタイムに確認したいとの要求もあり、必要な場合には、動的な共有を併用する必要がある。

### 3. マルチプラットフォームへ対応

従来の集中型や分散型では、OSあるいはウィンドウシステムに依存する、入力イベントや出力を分配するために、複数の異なったプラットフォーム間での実行は困難であった。中間方式では、システムに独立な抽象度の高い中間のプロトコルで通信を行なうために、異機種混在環境での実行が可能とする。また、グループウェア自体をスクリプトで記述する方式は、そのプログラムの可搬性を高めている。現システムの場合、UNIX/X-WindowとWindows-NTの2つの環境をサポートしている。

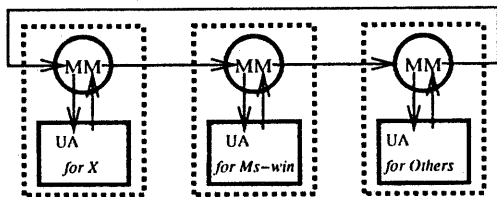


図 5: マルチプラットフォーム対応

## 7 おわりに

同期型グループウェアのアーキテクチャとして中間型を採用することにより、応答性の改善による低速なWANへの対応、操作自由度の向上、および他プロセスとのデータ交換を実現した。また、異機種プラットフォームへの対応を容易にした。これは、以前のシステムがX window systemをベースにした構成であったため、主にUNIXを開発開発環境として利用するチームでのみしか利用できず、PCをベースにしたチームにおいても利用したいとの声に応える結果となった。

今後の課題としては、メッセージマネージャの階層化などReGAPIの拡張、他のグループウェア開発への適用の推進と、TCSで記録したデータの情報管理を考えている。

## 参考文献

- [1] 貫井 春美 他: ソフトウェア分散開発支援システム  $d^2$ , 情報処理学会研究報告, volume 92, pp. 41-48. 情報処理学会グループウェア研究グループ, Dec 1992.
- [2] 松尾 他: 電子メールにおけるエージェントシステム, 第43回全国大会講演論文集. 情報処理学会, 1991.
- [3] 太田 他: ソフトウェア分散開発支援向け電子会議システムの開発, 第44回全国大会講演論文集. 情報処理学会, 1992.
- [4] H.M. Abdel-Wahab and M.A. Feit: Xtv: a framework for sharing x window clients in remote synchronous collaboration. In *In Proceedings of IEEE TriComm '91: Communications for Distributed Applications and Systems*, pp. 159-167, 1991.
- [5] 村永, 阿久津, 児玉, 今井, 守安: Open group media -個人用アプリケーションを共有するグループウェア・フレームワーク, 情報処理学会グループウェア研究会, volume 34, pp. 121-128, Oct. 1993.
- [6] 堀川 桂太郎, 桑名 栄二: 共有透明ボードシステムの試作 -cognet share-, 情報処理学会グループウェア研究会, volume 34, pp. 13-19, Oct 1993.
- [7] 阿部, 前野, 他: マルチメディア分散在籍会議システム (mermaid) を利用したグループアプリケーションの分散協調制御方式の提案, 情報処理学会論文誌, Vol. 34, No. 6, pp. 1406-1416, June 1993.
- [8] 中山 良幸, 森 賢二郎, 他: 多者間電子会議システム associa, 情報処理学会誌, Vol. 32, No. 9, pp. 1190-1199, Sep. 1991.
- [9] J.K. Ousterhout: Tcl: An embeddable command language, In *the Proceedings of the 1990 Winter USENIX Conference*, 1990.
- [10] J.K. Ousterhout: An x11 toolkit based on the tcl language. In *the Proceedings of the 1991 Winter USENIX Conference*, 1991.