共有空間通信を用いたグループウェアの構築

増井俊之　花田恵太郎　音川英之

シャープ株式会社 技術本部

情報技術研究所

{masui,hanada,otokawa}@shpcsl.sharp.co.jp

現存のグループワーク支援ツールの多くは表現力が足りなかったり特定のシステムを想定して作られていたりしており適用範囲が狭く柔軟性に乏しい。本論文では広い範囲の CSCW システム及びユーザインタフェースソフトウェアの基盤として使用可能な「共有空間通信モデル」を提案し、グループワークへの適用例を示す。本モデルは同期型 / 非同期型 CSCW の両者に対応しており、異なる言語やプラットフォームで共通に使用することができる。

Groupware based on Shared-Space Communication

Toshiyuki MASUI, Keitaro HANADA, Hideyuki OTOKAWA

Information Labolatories

Corporate Research and Development Group

SHARP Corporation

{masui,hanada,otokawa}@shpcsl.sharp.co.jp

We propose the *shared-space communication model* for CSCW applications. Unlike most of the current groupware construction tools which are limited in their expressive power and flexibility, shared-space communication model is simple, flexible, and able to support both synchronous and asynchronous group communication under different platforms and languages. We show how a CSCW system can easily be built using the model, by making a prototype teleconference system on workstations connected with a TCP/IP network.

1 はじめに

近年計算機のグループワーク (Computer Supported Co-operative Work, CSCW) への応用の有効性が認識されつつあり、グループワークを電子的に支援する各種の電子会議システムや編集システムが提案されており商用となっているものも多い。また CSCW システムを作成するためのツールも数多く提案されている。

グループワークを支援するソフトウェアではシングルユーザのアプリケーションでは必要のない様々な機能が要求される。例えば複数ユーザが同じデータを見たり操作したりする機能が多くの場合必要であり、その場合ユーザ間の制御の移動を適当に処理するための機能が必要である。またアンケートや投票機能などの機能が必要となる場合も多い。既存のユーザインタフェース構築支援システムはこれらをサポートしていないため上記のような機能を有する CSCW アプリケーションプログラムの作成を支援するためのツールキットが近年多数提案されている [2] [8] [13] [20] [22] [25] [26]。しかしこれらのシステムは以下のいずれかの問題点を持っている。

- X ウィンドウなど特定のシステムの機能に依存しており汎用性に乏しい
- 特定の対話モデルに限定されている
- 同期型 CSCW と非同期型 CSCW の両者に対応していない
- 既存のインタフェースツールとの整合性がわるい
- 一対一の通信を行なっているため相手について詳しい知識がないと通信ができない
- 通信の制御のような低いレベルから CSCW のアプリケーションの制御までの広い範囲をカバーしようとしている
- シングルユーザ向けシステムを複数ユーザ用に拡張しているため無理がある

本論文では、広い範囲の CSCW システム及びユーザインタフェースソフトウェアの基本構造として使用できる「共有空間通信モデル」を提案し、本モデルにより以上のような問題が解決されることを適用例を通じて示す。

2 共有空間通信による CSCW

我々は Linda [5] [7] を拡張した共有空間通信モデルに基づいたユーザインタフェースソフトウェアの構築手法を提案している [15][16]。本節では共有空間通信モデルについて簡単に解説を行ない、本モデルが通常のシングルユーザアプリケーションだけでなく複数ユーザのためのグループワークアプリケーションの構築にも有効であることを示す。

2.1 ユーザインタフェースにおける並列処理

近年の高度なユーザインタフェースソフトウェアでは並行処理機能が必須となってきている。マウスとキーボードのような複数の入出力装置を使用するプログラムでは各入出力装置に対し別々のプロセスを割り当てて処理を行なう方が簡単であるし、テキスト入力枠やボタンなどのインタフェース部品が並んだグラフィックインタフェース画面においては、ユーザが任意の順番で部品を操作可能にするためにはそれぞれの部品を並行に動作させなければならない。またユーザをプログラムと独立に動くプロセスと考えると都合が良い場合もあるし、 UIMS の大きな目標であるアプリケーション部とユーザインタフェース部の分離にも有効であるため近年のユーザインタフェース作成ツールのほとんどが明示的または暗黙的に並列処理をサポートしている。

明示的にユーザインタフェース専用の並列処理言語を使用した例として、 CSP[11] を使ったインタフェース記述言語 Squeak[4], Newsqueak[21] や、 ERL というイベント処理言語を使用した Sassafras UIMS [10] などがある。

暗黙的に並列処理を行なっているものとしては制約記述言語やユーザインタフェースツールキットがあげられる。近年のグラフィックインタフェース作成システムでは、グラフィックオブジェクトの間に制約を設定しオブジェクトの位置などの変化により自動的に別のオブジェクトが更新されるというものが数多く提案されているが [3] [9] [12] [23] [24]、オブジェクト間の制約は暗黙的な並列プロセスが制約の数だけ存在することと同等である [1]。

一方 X ウィンドウや NeXTstep などのウィンドウシステムでは各種のインタフェースツールキットと呼ばれるライブラリ群がインタフェースプログラムの作成に使用されている。インタフェースツールキットとはテキスト入力枠やボタンなどのグラフィックインタフェース部品の外見及び動作をライブラリとして定義したもので、各部品に対しマウスやキーボードなどから操作を加えると部品に対し定義された「コールバック」関数が呼び出されるようになっている。インタフェースツールキットではシステムが入力装置からの要求をとりまとめてツールキットに渡し、ツールキット側においてその要求がどの部品に対応するのか判断して各部品にイベントを配送するという構造になっているものが多い。この場合ツールキットのイベント配送システムがインタフェース部品の並行動作を支援していることになる。

以上のような並行処理表現は、特定の応用には便利であるものの、汎用のインタフェース記述方式としては不充分である。例えば制約解決システムは図形の配置やスプレッドシートのようなアプリケーションには便利であるがプログラムモジュール間の複雑な通信を表現することはむずかしい。 Squeak や ERL のような言語を使用すると明示的に並列動作の記述をきめこまかく行なえるという利点はあるが、並列処理だけのために特殊な言語を使用する必要があるためアプリケーションプログラムの開発の障害となる。インタフェースツールキットはグラフィックインタフェース作成に現在最も広く使用されているが、プログラムの制御移動の枠組が固定的で柔軟性に乏しく、部品は常にイベント配送システムから呼び出されるという形になるためプログラム構造が制約をうけてしまいアプリケーションを作成