２Ｋ−８

ＯＳ／２上の画面処理支援サブシステムの開発

苅田　茂
（株）東芝　青梅工場

１．はじめに

ネットワーク環境下で効率の良いシステムの運用するために、オンライントランザクション処理（以下、ＯＬＴＰ）を使うことは、一つの効果的な方法である。

ＯＬＴＰのホストマシンとしては汎用コンピュータやオフィスプロセッサの上位機種を使い、ワークステーションとして下位のオフィスプロセッサを使う従来の方法の他に、ワークステーションとしてパソコンを使ったシステムが多くなってきた。これは、

・パソコン用マイクロプロセッサの性能の向上
・高機能のパソコン用プラットホーム（ＯＳ、ウィンドウシステムなど）の登場

などが主な原因である。

米国マイクロソフト社とＩＢＭ社の開発したＯＳ／２　は仮想メモリ、マルチタスクなどの機能を持った、高機能の次世代のパソコン　ＯＳである。また、ＯＳ／２　は標準でプレゼンテーションマネージャというウィンドウシステムを持っている。

ここでは、プレゼンテーションマネージャ上で動作するアプリケーションの開発支援ツールとして画面処理支援サブシステムを開発したので、その技術的な側面について報告する。

２．アプリケーションモデル

どのようなアプリケーションであっても簡単に作成できる開発ツールであれば理想的であるが、現実的な考え方ではない。今回の開発では、ターゲットとなるアプリケーションの分野を限定した。オフィスプロセッサと連携したＯＬＴＰシステムの、ワークステーション側のプログラムの開発支援ツールの作成を目標としているので、データエントリ分野のアプリケーションの作成を支援することに的を絞った。

データエントリの分野はキーボード主体のオペレーションが中心であり、ＧＵＩとは相反する世界である。逆にいえば、プレゼンテーションマネージャの基本機能に欠けている部分でもある。

データエントリをおこなうオフィスプロセッサ上のアプリケーションの多くは、メニューによって階層化された同じような構造を持っている。そこで画面処理支援サブシステムを実現するために、

1)目的とするアプリケーションの論理的な構造をモデル化する。
2)このアプリケーションモデルを動かすサブシステムを作成する。
3)アプリケーションモデルに沿ってアプリケーションを構築する。

という方法をとった。

アプリケーションモデルには、以下に示す４つの階層を持つ木構造を採用した。

ＰＵ（Presentation Unit ）…
　アプリケーションの全体。木構造のルートになる。複数のＷＵを含む。

ＷＵ（Work Unit） …
　まとまった一つのデータ処理の単位で、複数のＳＣを含む。
　アプリケーションのデータ構造はＷＵを単位にして管理をおこなう。
　（たとえばデータの保存処理、復元処理、登録処理といった処理単位を、それぞれ一つ一つのＷＵと考える。）

ＳＣ（Screen）…
　ウィンドウの中に表示される一つの画面を示す。複数のＦＬを含む。

ＦＬ（Field） …
　画面上に配置される入出力単位。

ここで、実際に画面上に表示されるのは、ＳＣ、ＦＬであるが、これ以外の管理単位としてデータ構造を管理するＷＵを設けることにより、複数のウィンドウが一つのデータ構造を共有することが可能になっている。
（図１、図２参照）

A Design of Display Interface Development Tool on OS/2 Presentation Manager
Shigeru Kanda
TOSHIBA CORPORATION OME WORKS

3．ウィンドウによる管理

上記の制御単位を実現する方法として、一つ一つの制御単位をウィンドウとして作成した。ＰＵ、ＷＵなど画面上には表示されない制御単位については、プレゼンテーションマネージャに用意されている「オブジェクトウィンドウ」と呼ばれる機能を利用した。これは画面には表示されないが、プログラミング上は普通のウィンドウと同じように、メッセージのディスパッチを受けることのできる特別なウィンドウである。

またＦＬについては、取り扱うデータの種類ごとに別のＦＬを用意する方式とした。たとえば、

- データエントリのための細かいデータチェック機能を持ったＦＬ
- ボタンを取り込んだＦＬ
- イメージデータを取り扱うＦＬ
- 表データを取り扱うＦＬ

などを別々のＦＬとしてに用意する。このとき、ＳＣとのインターフェイスは全ての種類のＦＬについて標準化することが重要である。

このようなモジュールの構成にすると、新しいＦＬを追加していくことで、機能拡張に柔軟に対応できるという長所がある。

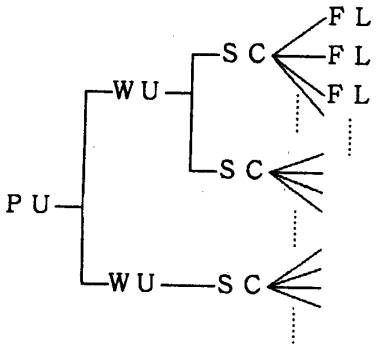

図１

4．結果と今後の課題

データエントリのアプリケーションでは、一つのＦＬの入力がおこなわれる毎に、できるだけ細かいデータチェックをおこない、不正なデータの入力では先に進めない、といった入力処理をおこなう。

これに対してＧＵＩでは、マウスを使って順不同に、分かる部分からデータ入力してゆき、一つの画面の入力が完了した時点でデータチェックをおこなうという処理方法をとる。

このような差は、入力するデータに対してどの程度の知識を持っているオペレータを想定しているかの違いによる。すなわち、キーボード主体のデータエントリの場合は、データの内容についての知識を持たないオペレータが単純入力をおこなう。ＧＵＩでは、コンピュータについては詳しくなくても、データの内容については知識のある者がデータを入力、修正することを前提にしている。データエントリが入力速度と正確さを重要視しているのに対して、ＧＵＩでは操作性と曖昧さを重要視している。

今回採用したアプリケーションモデルでは、マウスによって順不同にＦＬを選択して入力した場合に、キーボード主体のデータエントリでの求められる入力データの細かいチェックを実現することが難しい。今後はモデルの見直しなどによりデータチェック処理の強化を進めてゆきたい。

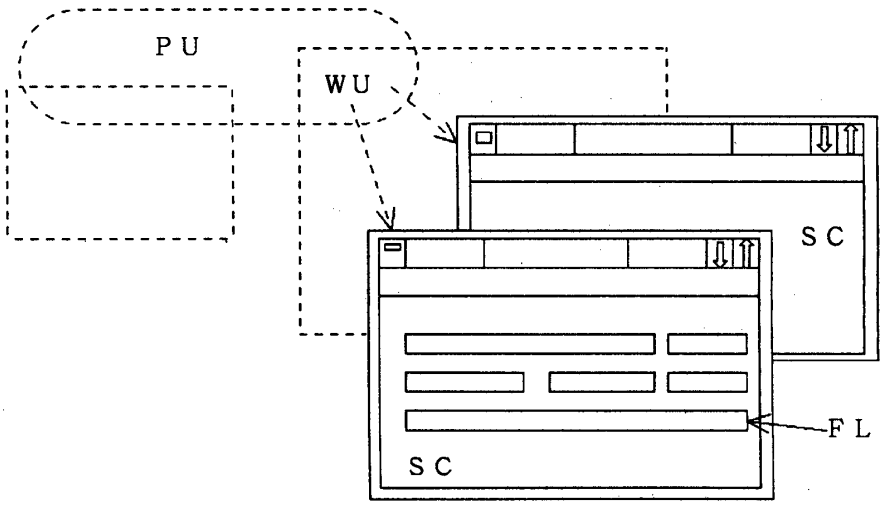

図２