

OS/omicon V4 における電紙のプログラミングモデル

3H-4

早川栄一, 森永智之, 加藤泰志, 並木美太郎, 高橋延匡

(東京農工大学 工学部 電子情報工学科)

1. はじめに

手書きを含めたパターンは、データの持つ多様性、多義性の性質を考慮する必要がある。このパターンをプログラミングする場合、従来のストリームのような単純なデータモデルでは、データをストリームに直すオーバーヘッドが大きく、効率よくプログラムを作成することは難しい。また、共有を中心としたプログラミングモデルもあるが、データ保護の問題がある。

我々は、これに対して紙を仮想化した電紙のデータモデル[1]を提案してきた。本報告では、OS/omicon V4 の中心となるデータモデルである電紙のプログラミングモデルについて述べる。

2. 「電紙」

「電紙」は、紙を仮想化したデータモデルである。手書きインタフェースのデータとペン操作に対して、多様性、多義性を持つ。

多様性に対して、通常の紙に性質にあたる「属性」を定義する。属性の型としては、文字コード、筆点列、ビットマップ、リンクの基本型と、ユーザ定義型の二種類を提供する。ユーザ定義型の処理部はアプリケーション(AP)プログラマが記述する。多義性については、属性間の変換機構を提供し、これを AP レベルで定義できるようにする。これを「抄紙」と呼ぶサーバによって提供する。

3. 設計方針

このような電紙を作成するプログラミングモデルの方針を、我々は次のように定めた。

- (1) データとそのアクセスメソッドがシステムで一貫していること
- (2) 保護機構を用意すること
- (3) アクセスメソッドを動的に拡張できること

4. プログラミングモデル

AP プログラムの実行図を図1に示す。

AP は、初期化と実行関数群から構成される。初期化部は、抄紙サーバへの実行関数群の登録と、データの初期化を行う。データ入力やタスク間通信、例外処理などのイベントが発生すると、関数がサーバ内部のスレッドで実行する。実行速度を向上させるために、関数の集合体として記述する。

AP は、抄紙サーバ内から呼び出され、抄紙サーバ内部のタスク上で実行する。抄紙サーバは、イベントに応じた複数のタスクを実行する。その間のタスク間通信についても、関数呼出しのインタフェースとして実現する。この場合、相手の関数名と、データアクセス権のリスト、引数を通知する。実現上は、リンケージテーブルへのアクセスが、ゲートコールの形で引き継がれ、サーバを経由して、対象のプログラムへのイベントとして生成される。

電紙の型(ユーザ型)を新しく定義する場合は、まず型とそれに対応する実行関数を定義する。次に、名前空間の中にその処理関数を置き、検索規則に関数の位置の定義を追加する。次にプログラムが使用する場合、ダイナミックリンクがその位置を探して、関数名と実体との結合を行う。

このようなモデルでは、関数呼び出しのために、実行速度は速いが、関数名と実体との対応を取る必要がある。これは、手書きデータの多義性からも生じる。また、データを共有することから、保護を行う必要がある。

電紙データはすべてのアプリケーション(AP)で共有する。共有の場合、データ保護の問題が重要になるが、これに対して、データに対する

操作権を定義し、AP 間でそれを受け渡す機構を提供する。

5. 属性

属性は、資源に対してそのアクセスメソッドを定義するものである。図2に属性の全体構成について示す。

属性は、資源に対する処理手続き、データと検索規則という3つを構成要素とする属性ごとの手続きやデータは、属性に対応した名前空間に存在する。また、検索規則は、ダイナミックリンクのリンケージフォールト時に検索を行う名前空間の中の一つの実体を示している。属性を定義することは、属性に合った手続きやデータを定義し、検索規則中に追加することと同じである。その規則上にはない場合は、別の規則を用いる。

名前空間は木構造になっていて、ルートからたどれば一意に決まるので、システム全体の資源を一意に指定することができる。

システムの属性名は、デフォルトと絶対という二つの表現がある。通常は、検索規則を用いたデフォルトが使用される。属性を一意に指定する場合は、'\$'で区切られた属性名を用いる。これは、プログラミング言語中の文字列リテラルもしくは、識別子としても扱うことができる。

6. 操作権

データに関するアクセス権は、操作権という単位で管理する。操作権は、セグメント単位のデータのアクセス権のリストである。操作権によって、セグメントに対するアクセス同期や排他を可能にする。また、操作権を他のタスクに渡すことで、共有データの流れを制御できる。

操作権は、マイクロカーネルが提供する保護空間[2]の機構を用いる。保護空間は、タスクとは別に設定できることから、セグメント単位での操作権の設定、受け渡しが可能になる。

7. おわりに

本稿では、OS/omicon V4 における「電紙」

プログラミングモデルについて述べた。抄紙サーバは、現在 Intel x86系と HP PA-RISC 上に実現中である。今後は、AP 作成による本モデルの評価と本モデルのオブジェクト指向による再モデル化を行う予定である。

参考文献

- [1] Hayakawa et.al : Basic Design of SHOSHI Operating System That Supports Handwriting Interfaces, Trans. IPSJ, Vol.35 No.12, 1994.
- [2] 森永他 : OS/omicon V4 のためのマイクロカーネルの設計, 情処学コンピュータシステムシンポジウム, Vol.94 No.10, 1994.

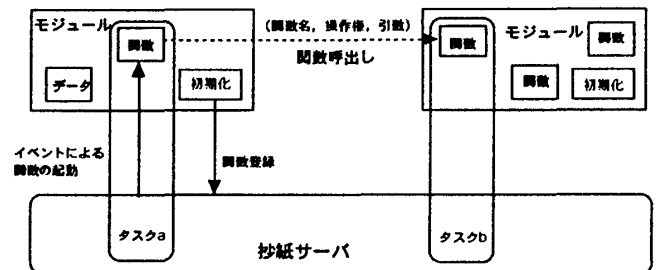


図1 APプログラムの実行図

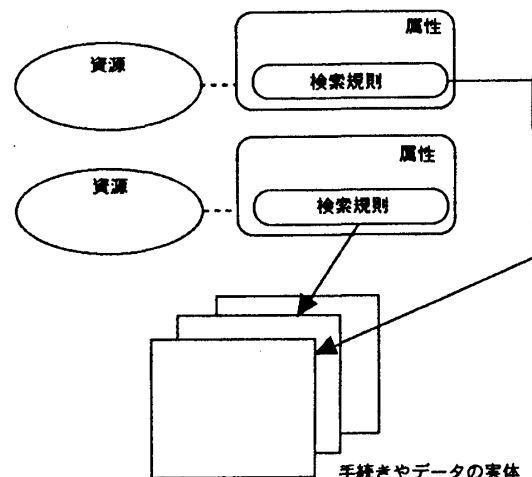


図2 属性の全体構成