OSIトランスポート・プロトコルへのAdaの適用と評価

2U-4

堀内 浩規　長谷川 亨　加藤 聰彦　鈴木 健二
国際電信電話株式会社 研究所

1. はじめに

近年、異システム間通信の実現にむけてOSI(開放型システム間相互接続)の標準化が進み、それに基づく実装が各国で進められている。筆者等は、大規模で信頼性の高いソフトウェアの開発に適したプログラミング言語Adaを用いたOSIプログラムの実装方法について先に提案している[1]。

本稿ではその方法に基づいてOSIトランスポート・プロトコル(TP)クラス0[2]を実装したので、その結果と評価について報告する。

2. AdaによるTPクラス0プログラムの構成

AdaでOSIプログラムを統一的に作成するために以下の基本方針を立てた[1]。

(1) 複数のレーヤを1つのプログラムで実現する。

(2) レーヤ毎に、1つのコネクションに対して、プロトコル手順を実行するコネクション・タスクとバッファリング用のキュー・タスクを設け、さらにこれらを一元的に管理するマネジメント・タスクを用意する。コネクション・タスクとキュー・タスクはコネクションの確立・解放に応じて生成・消滅する。

(3) ビジーウェイトによるスループットの低下やデッドロックが起こらないタスク構成をとる。

このような基本方針により、筆者らがVAX11/780(OSはVMS)上で開発済みのOSIプログラム[3,4]の内、TPの部分をTPクラス0 Adaプログラムで置き換えるように実装した。

2.1 タスク構成

作成したTPクラス0プログラムのタスク構成を図1に示す。基本方針(2)で述べた3種類のタスクは、図のT__CON, T__QUE, T__MANに対応する。この外に、メールボックスを介して既存のOSIプログラムからのインタフェース・プリミティブを読みこむために、インタフェース・タスクST__INとTN__INを用意しており、またTPクラス0で使用されるTS1とTS2タイマを実現するためにタイマタスクTIMERをT__CONの子タスクとして定義した。

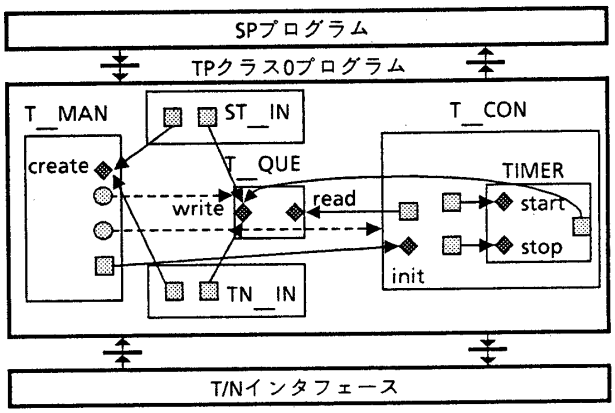

図1 TPクラス0プログラムのタスク構成

2.2 T__MANの処理

T__MANではコネクション確立要求/指示に対してエントリcreateが呼ばれ、リファレンス等を選択した後T__CONとT__QUEを生成させ、T__CONのエントリinitを呼びリファレンス等を与える。

2.3 T__CONにおけるプロトコル手順の実現

T__CONでは状態遷移表に従ってプロトコル手順を実行している。状態と入力に対しては列挙型を定義し、状態遷移表については、TPクラス0では規模が小さいため、状態と入力に対するcase文を用いてプログラムとして記述した。

2.4 T__QUEの処理

T__QUEでは、インタフェースプリミティブに対するキューを設けており、キューが空でなければエントリreadにおいて、キューが一杯でなければエントリwriteにおいてaccept文を実行する。

2.5 既存システムとのインタフェース

VAX/VMSではメールボックスの入出力を行うシステムサービスを定義している。VAX/Adaではシステムサービスを STARLETと呼ぶパッケージにまとめており、Adaプログラムで容易に使用できる。しかしVMSでは、Adaプログラムが複数のタスクを含む場合でも1つのプロセスとして実現して

Applicability of Ada to OSI Transport Protocol
Hiroki HORIUCHI, Toru HASEGAWA, Toshihiko KATO, Kenji SUZUKI
KDD R & D Laboratories

いるため、1つのタスクの中でシステムサービスを実行すると、その終了を待っている間はプロセス自身の実行が中断されてしまい、他のタスクの動作が停止してしまう。VAX/Adaではこれを回避する同期式システムサービスをパッケージTASKING_SERVICESに用意している。本プログラムではインタフェース･タスクST_INとTN_INで、この中のメールボックスからの入力待ちを行うシステムサービスTASK_QIOWを用い、それぞれが隣接レーヤからの入力待ちの状態でも、他のタスクが動作するプログラム構成としている。

2.6 インタフェース･タスクの処理

ST_INとTN_INでは、受信したインタフェース･プリミティブを、最初のコネクション確立に関する場合はT_MANに、その他の場合はコネクションに対応するT_QUEに渡す。そのため、ST_INとTN_INでは、各トランスポートコネクション(TC)に対して、TCと対応するネットワークコネクションの識別子及びT_QUEのアクセス値を関連づける管理テーブルが必要となる。従ってST_INとTN_INはTCの確立･解放に伴いこれらの情報をT_MANまたはT_CONから受け取る必要がある。しかし、ST_INとTN_INはメールボックスからの入力を待っているために、この制御情報の受け渡しを行うためにランデブーを用いることはできない。そこで今回の実装では、T_QUEのアクセス値を登録するための生成制御情報と除去するための消滅制御情報を、インタフェース･プリミティブと同様なフォーマットを用いてメールボックスを介して渡している。

2.7 タイマの実現

TIMERは、T_CONがタイマを起動するときに子タスクとして動的に生成される。生成されるとエントリstartからタイマの時間、TS1かTS2かのタイマの種類及び対応するT_QUEのアクセス値を受け取る。時間の経過はdelay文により実現され、この間にエントリstopが呼び出されない場合は、タイムアウト制御情報がT_QUEに書き込まれる。この制御情報もインタフェース･プリミティブと同様なフォーマットを有している。

2.8 バイトの扱い

バイトのためのデータ型は、Adaでは標準的には用意されていないが、VAX/Adaではシステムに依存する特性の定義を含むパッケージSYSTEMの中でバイトやワード等のデータ型を定義している。プリミティブやTPDUのデータ型は、この中の型を用いて記述している。

2.9 無検査型変換

Adaは厳密な型チェックを行うことが特長となっているが、以下の二つの部分で無検査の型変換を行う必要があった。

①TSAPアドレスとNSAPアドレスの対応関係は、ファイルに文字列で格納した。一方プリミティブやTPDUのデータ型ではこれらをバイトの配列として扱う必要がある。従ってファイルから読み取ったアドレスをプリミティブやTPDUに代入するために文字型とバイトの間の変換を行った。

②生成制御情報としてT_QUEのアクセス値をプリミティブと同じデータ型で渡すため、T_QUEのアクセス値と2ワードの間の変換を行った。

2.10 パッケージの利用

各種データ型、パラメータコードの値、定数、TPDUのフォーマッティング、プリミティブ作成等の基本的な関数/手続きは、パッケージの中にまとめている。

3.結果と考察

今回の実装で得られた結果を以下に示す。

①プログラムの実効行数は3K行程度である。

②既存のセションプログラムと組み合わせてX.25網折り返しにより、80Kバイトのファイル転送実験を行った。TPDUサイズを128バイト、SPDUサイズを1024バイトとして、スループットクラス4.8Kbpsの回線上で、約4.2Kbpsのスループットを得ている。

③本実装では他言語とのインタフェースを用いず、TPクラス0を全てAdaで記述している。

④T_QUEはキューの長さを64としており、実行時に132Kバイトの記憶領域が必要となり、長さ節を用いて明示的に指定する必要があった。

⑤厳密な型チェックによりデバックが比較的容易であり、開発効率の良さを実感した。

4.おわりに

本稿では、AdaによるTPクラス0の実装及び通信実験の結果を述べた。今後とも引き続きAdaによるOSIプログラムの実装を行う予定である。最後に、日頃御指導頂くKDD研究所野坂所長、小野次長、浦野情報処理研究室長に感謝します。

参考文献 [1]: 加藤,堀内,長谷川,鈴木,"AdaによるOSIトランスポート･プロトコル･クラス0の実装,"第32回情処全大,6D-6 ,Mar., 1986

[2]: CCITT, Rec. X.214, X.224, Oct. 1984

[3]: 鈴木,加藤,"OSIトランスポート･プロトコルのインプリメントと製品検証,"情処学会分散処理システム研究会,22-9, May 1984.

[4]: 鈴木,加藤,"OSIセッションレーヤ標準のインプリメント,"情処学会分散処理システム研究会,24-4, Nov. 1984.