2U-3

OSIトランスポート・プロトコルへのSmalltalk-80の適用と評価

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

通信ソフトウェアの多様化・大規模化にともない、その設計の段階が重要となりつつある。特に、通信システムのユーザインタフェースや処理速度の高い通信プログラムの開発では、設計時において、開発効率の良い言語でのプロトタイピングによる評価が有効であると考えられる。

筆者等は生産性の高いオブジェクト指向言語Smalltalk-80[1]を通信ソフトウェアに適用することを検討しており、OSIトランスポート・プロトコル(TP)[2]のクラス0を実装した[3,4]。本稿ではその結果と問題点について報告する。

2. Smalltalk-80によるTPクラス0の実装

2.1 TPクラス0プログラムの構成

筆者等は先に、OSIの各レーヤの実装を統一的に行うために、Smalltalk-80を用いたOSIの全レーヤに共通なプログラム構成を提案し、それに従いTPクラス0プログラムを実装した[3,4]。

このプログラムでは、トランスポート・コネクション毎にプロトコル手順を実行する機能と、コネクションを一元的に管理する機能をオブジェクトに対応させる。コネクションに対応するオブジェクトはキューを介してインタフェース・プリミティブをやりとりし合いながら並列動作する。

この並列動作は、上位及び下位のキューからのインタフェース・プリミティブの処理の繰り返しを、それぞれプロセスとして実行させることにより実現される。但し、これらのプロセスはそのコネクションの状態を共有するので、状態変数を共有変数とする必要がある(図1参照)。

2.2 実装時の問題点

次に、前節のTPクラス0プログラムを効率良く実行させるため、実装上考慮した問題点について述べる。スループットを向上させるには、プロセス構成を最適にし、処理量が大きいと考えらるプロトコル・データ・ユニット(PDU)の処理の効率を高くする必要がある。これらに対して以下のような検討を行った。

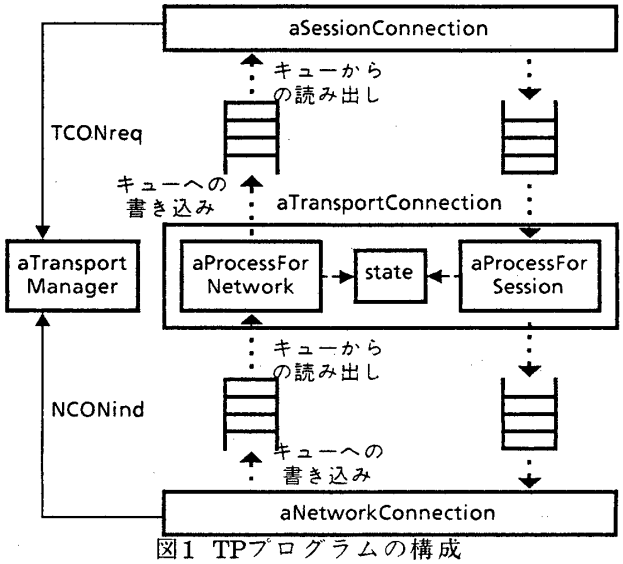

図1 TPプログラムの構成

2.2.1 プロトコルデータユニットの扱い

PDUの処理は各レーヤにおいて頻繁に行われ、その処理効率向上には、レーヤ間での無駄なコピーを省くことが重要である。PDUの無駄なコピーが起こり得る場合としては、①ヘッダの書き込みと削除、または②PDUのセグメンティング(分割)とアセンブリング(組み立て)の2つが考えられる。今回の実装では、PDUをユーザデータの前後にプロトコル制御情報を付加できるような構造のバイトの配列として扱うことにより、①の場合の無駄なコピーを避けた。

2.2.2 バイト列のコピー

Smalltalk-80でバイト列を1バイトずつコピーする処理は、効率が非常に悪く、そのためバイトの配列を一括してコピーするプリミティブなメソッドが用意されている。従って、PDUの全てのコピー又は部分コピーを行うメソッドは、全てこのプリミティブなメソッドを用いて実現することによりコピーの速度を向上させた。

Applicability of Smalltalk-80 to OSI Transport Protocol
Toru HASEGAWA, Toshihiko KATO, Kenji SUZUKI
KDD R & D Laboratories

2.2.3 プロセスの扱い

プロセス構成についてはこれまでに考察しおり、ビジー・ウェイトにならない構成を取っている[3,4]。プロセスが処理効率に影響を与える要因としては他に、プロセス切り替えの頻度があり、実際にプロセス切り替えを頻繁に行う場合とあまり行わない場合の2通りの方法でその影響を確かめた。

3. 通信実験

TPクラス0プログラムをXEROX AIW1121上に実装し、RS232C回線上及びイーサネット上で実際の通信を行うことにより、その正当性及び性能を評価した。そのため、RS232C回線上及びイーサネット上で動作する2種類のネットワーク・レーヤ・プログラムを作成した。ここで、ネットワーク・レーヤはX.25レベル3を簡易にした手順である。

3.1 RS232C回線上での通信実験

本実験ではAIW1121の持つ非同期のシリアルインタフェースを用いて、先にパソコン上に作成したTPクラス0プログラム[5]との間の通信を行い、正常な動作を確認した。このネットワーク・レーヤでは、NPDUを直接RS232C回線上で送受し、NPDUの区切りはベーシック手順に基づくブロック同期により行っている。

3.2 イーサネット上での折り返し通信実験

TPクラス0プログラムのスループットを測定するため、イーサネット上での転送実験を行った。本実験では折り返し通信を行うためにXNS(Xerox Network Systems)のエコープロトコルを用いた。ネットワーク・レーヤはエコープロトコルにインタフェースし、NPDUをエコーパケットにのせ、コネクションの識別はエコーIDにより行っている。

TPクラス0上で、TSDUサイズ:1024バイト、TPDUサイズ:512バイト、NPDUサイズ:512バイトの条件で、20Kバイトの情報の折り返し転送を行い約7.2Kbpsのスループットを得た。

3.2.1 セグメンティングの影響

TSDUサイズ1024バイト固定でTPDUサイズを512、256、128バイトと変化させることにより、スループットは約7.2Kbps、約5.3Kbps、約3.6Kbpsと低下し、セグメンティングが頻繁に行われる程スループットが低下した。また、TSDUサイズ128バイト、TPDUサイズ128バイトで、スループットは約2.0Kbpsであり、トランスポート・レーヤでセグメンティングを行う方が行わない場合よりスループットは良くなった。この結果は、先にC言語で実装した場合の結果とほぼ同じ傾向を示している。

3.2.2 プロセス切り替えの頻度の影響

プロセス切り替えの頻度の多い場合として、インタフェース・プリミティブの処理が終了する度にプロセスを切り替える方法を、少ない場合として、キューが空になるまでプロセスを切り替えない方法を対象として、2通りの実験を行った。その結果、前者は後者に比較して、約30~40%程度スループットが低下した。他のプログラム言語と同様にSmalltalk-80においても、プロセス切り替えのオーバーヘッドが大きいという結果が得られた。

4. 結果と考察

提案した構成方法によりTPクラス0及びネットワークレーヤのプログラムを容易に作成することができ、本構成方法の有効性及び他のレーヤへの適用性の見通しを得た。以下に、通信プログラムにSmalltalk-80を適用した結果について考察する。

4.1. プログラムの作り易さ

TPクラス0のソースコードは約800行であり、筆者らが先に作成したC言語のプログラムと比較して非常に少なくてすみ、それに伴い開発効率も高くなった。また、2種類のネットワーク・レーヤ・プログラムをあまり変更することなく作成することができ、Smalltalk-80のプログラムのモジュラリティの高さによる変更の容易さが確かめられた。

4.2 プロトタイピングへの適用性

Smalltalk-80により作成した通信プログラムでは、3.2節の実験結果のように、通信ソフトウェアに重要なプロトコル機構やプロセス切り替え等のシミュレーションが容易であり、かつそれらの結果は他の言語による場合と同様の傾向を示している。従って、Smalltalk-80によるプロトタイピングの結果を、C言語等の既存言語によるプログラム開発に反映させることができると考えられる。

5. おわりに

本稿では、Smalltalk-80をOSIトランスポート・プロトコルの実装に適用し、その通信プログラムへのプロトタイピングの見通しを得た。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]: Goldberg, A. et.al.,"Smalltalk-80: the language and its implementation",Addison-Wesley, 1983.

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

[3]: 加藤,長谷川,鈴木,"オブジェクト指向言語Smalltalk-80によるOSIトランスポート・プロトコルの実装",61年度前期情処全国大会

[4]: 長谷川,加藤,鈴木,"Smalltalk-80によるOSIトランスポート・プロトコルのインプリメント",情処学会マルチメディアと分散処理研究会,Jul,1986.

[5]: 加藤,鈴木,"パソコンへのOSIトランスポート及びセションプロトコルの移植",61年度前期情処全国大会