

OSI TP (トランザクション処理) プロトコルソフトウェアの設計と評価

杉 山 敬 三[†] 小 花 貞 夫[†] 鈴 木 健 二[†]

近年、複数の異機種システム間で処理の一貫性を保って分散処理を行う場合に重要となる OSI (開放型システム間相互接続) の TP (トランザクション処理) が注目されている。TP は、トランザクション処理を行うための応用層のプロトコルを規定しており、データベースアクセスやネットワーク管理など種々の応用から利用される。これらの TP を利用する応用では、処理の対象となる資源の種類や使用する応用プロトコルが異なる。このため、各種の応用から利用できる TP プロトコルソフトウェアを実装するには、応用が扱う資源の種類に依存せず、かつ応用が使用する各種応用プロトコルと組み合わせて動作可能であるという汎用性が重要となる。本論文では、TP プロトコルソフトウェアの実装において、このような汎用性を実現するための資源の管理方法とソフトウェアの構成法を論じた。資源の管理方法については、TP のプロトコルモジュール内に利用者 ASE (応用サービス要素) に特化した処理を排除できる点で TPSU (TP サービス利用者) が資源を管理する方法が有効であることを示した。ソフトウェア構成法については、ALS (応用層構造) における SACF (単一アソシエーション制御機能)、MACF (複数アソシエーション制御機能) ならびに各種 ASE をソフトウェアモジュールとして実現する際の機能分担を明確にした。上記の資源管理の方法とソフトウェア構成法に基づいて設計したソフトウェアを汎用性や処理効率等の点から評価し、本手法により実用的レベルの処理効率で各種応用から汎用的に利用できるソフトウェアを実現できることを示した。

Design and Evaluation of OSI TP (Transaction Processing) Protocol Software

KEIZO SUGIYAMA,[†] SADA OOBANA[†] and KENJI SUZUKI[†]

OSI TP is one of OSI application layer protocol for the distributed transaction processing. It is used in conjunction with several applications such as database access, network management and so on. In order to cooperate with those applications, TP protocol software shall meet the applicability that handles various kinds of resources and different application contexts. We point out that the local resource management and software configuration of TP protocol software are important factors to meet such applicability. As for local resource management, we compare following two cases. One is that management entity is in TPSP (TP Service Provider) and the other is that management entity is in TPSU (TP Service User). We propose the latter because the processing dependent on resources can be excluded from TP protocol software. As for software configuration, we introduce software modules for MACF (Multiple Association Control Function) and SACF (Single Association Control Function) based on ALS (Application Layer Structure). MACF module further separates into common MACF and application-specific MACF to enable application protocols to use common MACF. SACF module has the routing function of service primitives and concatenation/separation function of different ASE's APDU (Application Protocol Data Unit) so as to reuse ASE modules. We can show the applicability with practical performance through the implementation of software.

1. はじめに

OSI (開放型システム間相互接続) の標準化が進捗し、FTAM (ファイル転送、アクセスと管理) や MHS (メッセージ通信処理システム) といった OSI プロトコルを使用する通信システムが稼働し始めている。実

際の応用業務では、これらバッチ型の通信のみならず、商品の受発注や予約業務などの会話型の通信も必要とされる。このような通信形態はトランザクション処理と呼ばれ、オンライン業務の大半を占めると言われている。そこで、ISO (国際標準化機構) では、OSI の応用層のプロトコルとしてトランザクション処理の標準¹⁾ (OSI TP: 以下 TP と呼ぶ) を作成している。トランザクション処理では、複数のサイトに存在

[†] 国際電信電話株式会社研究所
KDD R&D LABORATORIES

するデータの同期更新や、複数のユーザが同一データにアクセスする際の同時性、障害からの回復などの機構が必要となるが、TP はこれらを実現するためのプロトコルを規定している。

従来のトランザクション処理システムは、メインフレーム上の DB/DC (データベース/データ通信) システムや OLTP (オンライントランザクション処理) 専用マシンのようにデータベースと組み合わせて実現されてきた。しかしながら、トランザクション処理自体は、データベースに限らず種々の応用に一般的に適用できる重要な処理の概念である。特に TP は、RDA (遠隔データベースアクセス) を始めとして、ネットワーク管理や RPC (遠隔手続き呼出し)、EDI (電子データ交換) など、分散処理を指向した各種の応用と組み合わせた利用が予想される。これらの応用では、ファイルやデータベースなど処理の対象となる資源の種類が異なり、また使用する応用プロトコルも、例えば RDA (データベースアクセスの場合) や CMIP (ネットワーク管理の場合) などと異なる。このため、各種の応用から利用できる TP プロトコルソフトウェアを実装するには、応用が扱う資源の種類に依存せず、かつ各種応用プロトコルと組み合わせて動作可能であるという汎用性が重要となる。TP プロトコルの実装については、これまでにも、モジュール化を考慮したソフトウェアの設計や試作^{2),3)}を行った例が報告されているが、これらはソフトウェアの実装、移植および保守の容易さを目的としたものであり、上記のような汎用性を目的としたものではない。

本論文では、各種の応用業務で汎用的なトランザクション処理機能を提供するための TP プロトコルソフトウェアの設計方法について論じる。ここではまず、資源管理方法やソフトウェア構成法がポイントとなることを示し、資源管理ではローカル資源の管理主体について、またソフトウェア構成では MACF (複数アソシエーション制御機能) と SAO (単一アソシエーションオブジェクト) の実現方法について検討する。さらに、これらの方法に基づき実装した TP プロトコルソフトウェアの機能について、資源管理とソフトウェア構成における汎用性や性能の観点から評価を行い、実用的レベルの処理効率で汎用的なソフトウェアを実現できることを実証する。

2. TP の概要

TP は開放型システム間で分散トランザクション処

理を実現するための機能を提供し、標準¹⁾では以下のように規定されている。

2.1 TP におけるトランザクションの概念

トランザクションとは、以下の四つの特性 (ACID 特性) を持つ処理の単位である。

- ・原子性 (Atomicity)
すべての処理を実行するかすべて実行しないかのどちらかである。
- ・一貫性 (Consistency)
制御対象のデータは一貫した状態に遷移する。
- ・独立性 (Isolation)
処理中の結果に外部からアクセスできない。
- ・耐久性 (Durability)
完了した結果は障害があっても変更されない。

TP では、TPSP (TP サービス提供者) が ACID 特性を保証する場合を提供者支援トランザクション、TPSU (TP サービス利用者) が保証する場合を応用トランザクションと呼ぶ。提供者支援トランザクションはさらに、トランザクションの連続性の観点から、一つのトランザクションの終了時に自動的に次のトランザクションが開始される連鎖トランザクションと、明示的に次のトランザクションの開始を指定する非連鎖トランザクションに分類される。

2.2 TP が提供するサービス

TPSU 間では、AEI (応用エンティティインボケーション) 間の通信路に相当するアソシエーション上にダイアログと呼ばれる関係を確立して通信を行う。ダイアログ上では、トランザクションの起動・コミット・ロールバックなどのコミットメントサービス、データ転送やエラー通知、処理の同期など表 1 に示すサービスが提供される。RDA など TP を利用する各種の応用の ASE (応用サービス要素) は U-ASE と呼ばれ、データ転送サービスである TP-DATA は U-ASE のサービスを総称したものである。また、複数システム上で協調動作する TPSU はダイアログツリーと呼ばれる木構造で関係付けられる。

2.3 資源とコミットメント制御

提供者支援トランザクションでは、トランザクションに関与する TPSU は、トランザクションツリーと呼ばれる木構造で関係付けられる。トランザクションツリー上のすべての資源は、コミットメントサービスによりトランザクション終了時に矛盾なくコミットまたはロールバックする。また、資源を要求されたシステムに存在するローカル資源は、データベースなど

TP を利用する応用に固有の制御対象データとトランザクション識別子などトランザクションの制御情報である原子動作データに分類される。

障害が発生した場合には、原子動作データを基に TP チャンネルと呼ばれる障害回復用のアソシエーション上で、資源の状態の一貫性を保つトランザクションリカバリを行う。

2.4 応用層構造における TP の位置付け

TP は、図 1 に示すように、ALS⁴⁾ と呼ばれる応用層の構造に関するモデルに基づいてプロトコルマシンを規定している。MACF は、ダイアログや TP チャンネルのアソシエーションへの割当てや、複数のアソシ

エーションにまたがるコミットメントの調停などの機能を持つ。SAO は、TP ASE, ACSE (アソシエーション制御サービス要素) ならびに一つ以上の U-ASE や、提供者支援トランザクションの場合に必要な CCR (コミットメント, 同時性および回復制御) ASE などの複数の ASE と、ASE 間のインタラクションの調停機能をモデル化した SACF (単一アソシエーション制御機能) から構成される。TP ASE の提供するサービス (NN サービス) は、MACF が唯一の利用者となる。

3. 汎用 TP プロトコルソフトウェアの設計における課題と解決方法

トランザクションは ACID 特性を保証する重要な概念であり、その応用は広範である。例えば、銀行のオンライン処理や座席予約などの各種予約業務処理、分散データベースアクセスに加えて、

- RPC における複数プロセス間の同期処理
- ファイル転送におけるファイルバージョン管理
- オフィス文書処理における同期文書編集
- ネットワーク管理における複数 MIB (管理情報ベース) にまたがる構成管理情報の更新
- EDI を用いた受発注業務処理

などの応用が挙げられる。また、ネットワーク管理とプログラムファイルのダウンロードのように、複数の応用をまとめてトランザクション処理を行う場合も考えられる。そのため、TP を利用するユーザの観点からは、図 2 に示すように、TP は様々な応用と組み合わせられて利用される共通的な要素と位置付けられる⁵⁾。

しかしながら、RDA で扱うテーブルや FTAM で扱う無構造ファイルなど応用ごとに固有の資源が存在するため、TP プロトコルソフトウェアを共通的に利用するには特定の応用が対象とする資源の種類に依存

表 1 TP の機能単位とそのサービス
Table 1 Functional Units and their services.

機能単位	サービス
カーネル	TP-BEGIN-DIALOGUE TP-P-REJECT TP-U-REJECT TP-END-DIALOGUE TP-DATA* TP-U-ERROR TP-P-ERROR TP-U-ABORT TP-P-ABORT
全二重型	(特定のサービスはない)
半二重型	TP-GRANT-CONTROL TP-REQUEST-CONTROL
ハンドシェイク	TP-HANDSHAKE TP-HANDSHAKE-AND-END TP-HANDSHAKE-AND-GRANT-CONTROL**
コミット	TP-DEFERRED-END-DIALOGUE TP-DEFERRED-GRANT-CONTROL** TP-COMMIT TP-CONTINUE-COMMIT TP-COMMIT-RESULT TP-DONE TP-COMMIT-COMplete TP-PREPARE TP-READY TP-ROLLBACK TP-ROLLBACK-COMplete
非連鎖トランザクション	TP-DEFER-NEXT-TRANSACTION TP-UNCHAIN-TRANSACTION TP-BEGIN-TRANSACTION

*:TPSUが応用に固有のASEのサービスを起動することを示す。

**半二重型機能単位が選択された時のみ使用される。

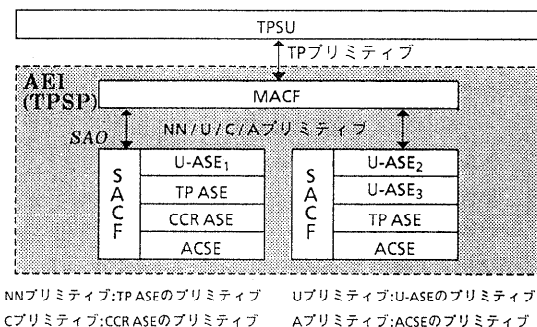


図 1 TP と ALS の関係

Fig. 1 Relationship between OSI TP model and ALS (application layer structure).

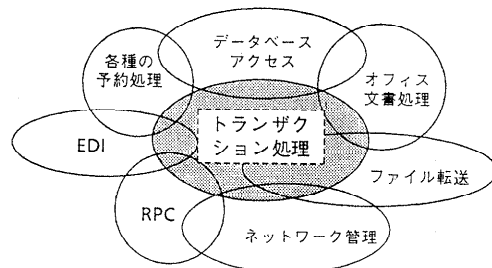


図 2 OSI における TP の位置付け

Fig. 2 Transaction processing in the OSI environment.

しないことが必要となる。また、TP を様々な応用プロトコルと組み合わせて動作させるには、使用する ASE の種類や各々の依存関係を規定する応用コンテキストの変更に柔軟に対処できなければならない。したがって、汎用的な TP プロトコルソフトウェアを設計する際には、次の 2 点が重要な課題となる⁹⁾。

- ・特定の応用が対象とする資源の種類に依存しないローカル資源の管理方法
- ・応用コンテキストの変更に柔軟に対応可能なソフトウェア構成

以下では上記の課題の解決方法について論じる。

3.1 ローカル資源の管理方法

ローカル資源のうち原子動作データは応用に依存せず TPSP で管理する必要があるが、制御対象データの管理は標準では TPSU と TPSP のどちらも可能としている。ここでは、汎用性の観点から各々の方式を比較する。

制御対象データの管理は、TP のコミットメント機能と U-ASE の機能の両方を用いて行う必要がある。RDA の場合、例えばテーブルのオープン・クローズや検索・更新等の操作は RDA のサービスと対応するが、コミットやロールバックなどのトランザクションの制御は TP のサービスに対応することになる。また、ユーザからのサービス要求を受け実資源にアクセスする資源管理モジュールは、資源ごとに作成する必要がある。実際、既存の資源管理モジュールである X/Open⁷⁾ のリソースマネージャにおいても、資源に依存しないインタフェース (XA インタフェース) から、資源に固有のインタフェースに変換しなければならない例が見られる。

したがって、TPSP が資源管理を行う場合、TP のプロトコル処理部分に U-ASE に特化した処理が必要となり、特定の応用に依存しない TPSP を実装することは困難となる。一方、TPSU が資源管理を行う場合には、TPSP はサービスプリミティブのマッピングや APDU (応用プロトコルデータ単位) の作成・解析を行うだけとなり、半二重やトランザクション終了中のデータ転送の禁止などのデータ転送に関する制限以外は U-ASE のプリミティブに関与しない。そのため、異なる資源を扱う U-ASE と組

み合わせることが容易であり、TP プロトコルソフトウェアの汎用性の観点からは TPSU に資源管理を行わせるのが望ましい。

3.2 MACF と SAO のソフトウェア構成

TP は従来の応用プロトコルと異なり複数アソシエーションにまたがる処理を行うため、ソフトウェア構成においては MACF と SAO の実現方法がポイントとなる。TP のプロトコル仕様では、MACF と単一アソシエーションに関する SAO の側面とに分けて記述されているため、標準との親和性や MACF と SAO の処理の独立性を考慮し、各々をモジュール化することとする。図 3 にソフトウェア構成を示す。

(1) MACF のモジュール化

ALS では規定されていないが、MACF の機能は、複数の AEI 管理や入出力管理といったどの応用にも共通な機能 (ここでは共通 MACF と呼ぶ) と、TP の状態遷移やトランザクション管理など応用に固有な機能 (ここでは特定 MACF と呼ぶ) に分類できる⁸⁾。そこで、他の応用プロトコルでも共通 MACF モジュールを利用可能とするため、共通 MACF と TP-MACF (TP 用の特定 MACF) を各々モジュール化することとする。

また、ダイアログの割り当てられていないアソシエーションを解放せずに保存するアソシエーションプールは現在 TP に特有の機能であるが、他の応用でも共通に使用できる。例えば RDA や CMIP ではアソシエーションの確立・解放処理は規定していないた

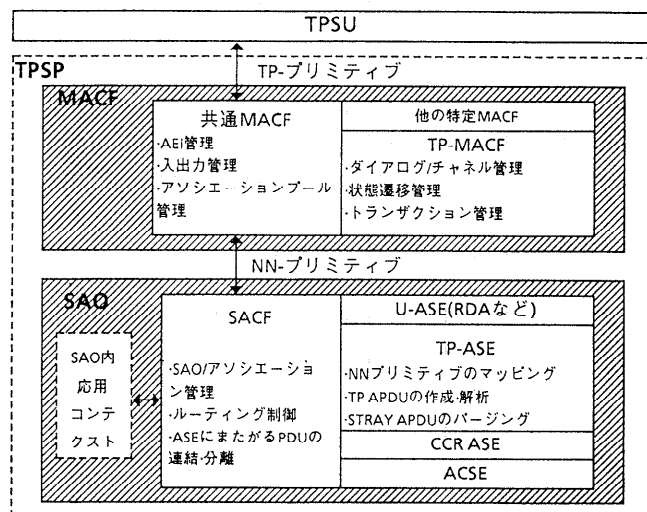


図 3 ソフトウェア構成

Fig. 3 Software configuration.

め、アソシエーションプールを適用することが可能である。そこで、アソシエーションプールの機能は共通 MACF に持たせ、特定 MACF がアソシエーションプールから応用コンテキストや機能単位などの条件を満足するアソシエーションを選択することとする。

(2) SAO のモジュール化

TP と種々の U-ASE の組み合わせを容易にするには、応用コンテキストに依存する部分を局所化し、異なる応用コンテキストに対しても ASE モジュールを再利用できる必要がある⁹⁾。応用コンテキストによるサービスマッピングの変更など ASE に閉じた機能は各 ASE モジュールが応用コンテキストに応じた動作をとるようにすればよい。そこで、ここでは複数の ASE に係わる機能、すなわちプリミティブを対応する ASE モジュールへ渡すルーティング制御と、異なる ASE 間で複数の APDU を一つの PSDU (プレゼンテーションサービスデータ単位) にマッピングする連結やその逆の処理である分離について検討する。

例えば ACSE からのアソシエーション確立指示プリミティブ (A-ASSOCIATEind) は、TP の場合は TP ASE に渡されるが FTAM では FTAM ASE に渡されるといったように、ルーティングは応用コンテキストで異なる。このルーティング機能を ASE モジュールに持たせると、応用コンテキストごとに ASE モジュールの変更が必要となり、ASE モジュールの再利用化が困難となる。また、FTAM のグルーピングや CCR の C-COMMIT-RI+C-BEGIN-RI のように、同一の ASE における APDU の連結・分離は、各 ASE が対応する状態遷移を持っているため、各 ASE に処理を行わせる。しかしながら、TP-BEGIN-DIALOGUE-RI+C-BEGIN-RI のような異なる ASE にまたがる APDU を連結・分離する場合、特定の ASE モジュールに処理を行わせると連結・分離のために特別なルーティング規則が必要となり、ルーティング処理が複雑となる。例えば、TP では TP・CCR・U-ASE の APDU が連結の対象となるが、CCR ASE のモジュールに処理を行わせた場合、TP と U-ASE の APDU を連結する場合にも CCR にルーティングしなければならない。

このような問題を解決するために、SACF に相当するモジュールを明示的に導入し、ALS では具体的に規定されていないトークンやプレゼンテーションコンテキストなどの ASE 間の共通情報の管理や、プリミティブのルーティング機能および異なる ASE 間の

APDU の連結機能などの応用コンテキストの制御を SACF モジュールで実現する。さらに、ルーティング情報を SACF モジュールに埋め込むと応用コンテキストごとにプログラムを変更する必要があるため、応用コンテキストのデータとして SACF のプログラムから独立させることとする。

4. TP プロトコルソフトウェアの設計

ここでは、3章の基本方針に基づいた TP プロトコルソフトウェアの設計について述べる。

4.1 設計方針

本設計に当たっては以下のような方針を立てた。

(1) TP プロトコルとして、設計時の最新版である DIS 第1版に準拠して表1に示す全機能単位を実装する。CCR は DIS 第3版¹⁰⁾に準拠する。また、U-ASE として、RDA ASE¹¹⁾と、任意のオクテット列を転送する無構造データ転送 ASE (UDT-ASE)¹²⁾を実現する。⁷⁾

(2) ACSE やプレゼンテーション層以下については、既に VAX 8700 (OS: VMS, 約 6 MIPS) 上に実装した IS 版のプログラム¹³⁾を使用する。

(3) SAO, MACF, TPSU のモジュールを各々独立のプロセスとし、その間でオクテット列で実現されるプリミティブを交換する。SAO のモジュールは、1 プロセスで複数のアソシエーションをサポートする。プロセス間通信には、VMS の提供するキューインタフェースであるメールボックスを使用する。

(4) プログラミング言語として C 言語を用いる。

4.2 MACF の実現方法

MACF の詳細な機能や SACF との関係、特にアソシエーションの確立・保存やリカバリ時の具体的な処理は ALS や TP の標準の規定範囲外であるため、以下のように実現した。図4に、MACF で管理する情報の関連を示す。

(1) 共通 MACF の機能

a) AEI 管理

AEI 識別子や AE タイトル、アソシエーション識別子など、各種応用で共通に使用される情報の管理を行う。これらの情報はアソシエーションごとに管理し、図4に示すように共通 MACF と特定 MACF 内の情報の関連付けを行い、特定 MACF から参照できるようにした。また、MACF と SAO の連携を図るため、共通のアソシエーション識別子を割り当てた。アソシエーション識別子は、起動側では MACF が奇

番号を、受諾側では SAO が偶数番を割り当て、アソシエーションごとにインクリメントするようにした。

b) 入出力管理

SAO および TPSU のプロセスとの通信と、特定 MACF モジュール間のプリミティブのルーティングを扱う。複数の TPSU プロセスをサポートするため、プロセス間通信のためのメールボックスの論理名と TPSU プロセスのプロセス識別子を対応させた。

c) アソシエーションプール管理

常に同一の相手と通信する場合などアソシエーションを固定的に利用する場合をサポートするため、AE タイトル等のアソシエーションの確立に必要な情報をファイルに保存し、起動時にそれを読み出してアソシエーションを自動的に確立する機能を設けた。また、無通信監視タイマを設け、アソシエーションが長時間使用されない場合には自動的にアソシエーションを解放するようにした。

(2) TP-MACF の機能

a) ダイアログ／チャンネル管理

TP-MACF では、図 4 に示すように TPSU タイトルや TP の機能単位などの TP に特有の情報をダイアログやチャンネルごとに管理し、共通 MACF のアソシエーション情報との対応をとるようにした。

TP-BEGIN-DIALOGUEreq プリミティブを受信した場合、まずアソシエーションプールを検索し、利用可能なアソシエーションが存在した場合にはそれを使用し、存在しなければ新しいアソシエーションを確立するようにした。また、アソシエーション確立中や、ダイアログ確立要求の衝突を避けるための BID 手順の実行中には、TPSU からの他のサービス要求

をキューイングする必要がある。本設計では TPSU とのプロセス間通信にキューインタフェースを使用するため、この間は他の要求をキューから取り込まないことで実現した。また、SAO へのキューイングは、TP-MACF 内にプリミティブのデータを保持することで実現した。

アソシエーションの解放処理は標準では規定されていないため、TP-END-DIALOGUE などダイアログ終了に関するプリミティブにアソシエーションの解放を指定する独自のパラメータを追加した。また、TP チャンネルに対応するアソシエーションは必要に応じて確立し、リカバリ処理の終了後に解放することとした。

b) 状態遷移管理

SAO と TPSU からのプリミティブを受信して、TP プロトコルで規定される状態遷移を行い、TP サービスと NN サービスをマッピングする。DIS 第 1 版ではプロトコルの状態遷移表が MACF と SAO に明確に分離されていない部分が存在するため、これらの状態遷移表を統合した状態遷移を TP-MACF に行わせた。これにより、MACF と SAO を独立のプロセスにしたことによる処理のすれ違いを防いだ。

状態変数やイベント、アクションには、FTAM など従来のプロトコルと異なり、各ダイアログに固有のものだけでなく、コミットメントの状態などノード全体に関するグローバルなものが存在する。グローバルな情報は図 4 に示すようにダイアログに固有な情報とは別に一元的に管理し、グローバルなアクションが発生した場合にはこの情報を参照して必要なノードに対してアクションを起こすようにした。

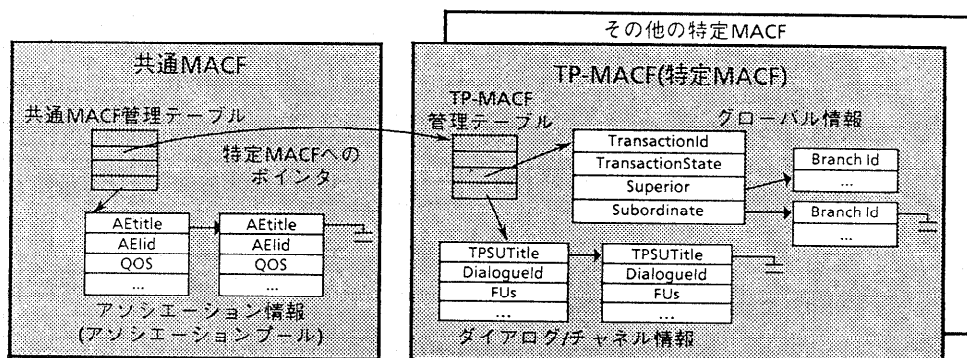


図 4 MACF における情報の関連

Fig. 4 Relationship between information in MACF.

c) トランザクション管理

トランザクション識別子などの原子動作データの生成・保存や、トランザクションリカバリといった OLTP のトランザクションモニタに相当する機能を果たす。障害には、アソシエーションは異常終了するが AEI は存続する通信障害と、AEI 自体が消滅するシステムクラッシュが存在するため、図 5 に示すフォーマットを用いてダイアログごとに原子動作データをログファイルとして保存するようにした。コミット処理の終了を示す FORGOTTEN ログレコードは、ログファイルの削除に対応させた。また、CCR でも原子動作データが規定されているが、これは TP の原子動作データの一部として包含できるため、処理効率を考慮して CCR ではログをとらないようにした。

通信障害の場合には、障害が発生したアソシエーションの情報を削除せずに主記憶上に保存し、その情報を基にアソシエーションを再確立することで、速やかにリカバリできるようにした。また、システムクラッシュの場合には、AEI の再起動時にログファイルを読み出して新たにアソシエーションを確立するようにした¹⁴⁾。アソシエーション確立後、ノードの状態を CCR プロトコルを用いて交換することでトランザクションリカバリを行うが、その後のダイアログ再開などの処理は TPSU に任せることとした。

4.3 SAO の実現方法

(1) SACF

a) SAO/アソシエーション管理

プレゼンテーションコンテキストと対応する ASE の識別子や同期点通し番号などの各 ASE で共通に使用される情報および SAO のインスタンスを管理する。これらの情報は、図 6 に示すようにアソシエーションごとに管理し、ASE モジュールからの参照および設定を可能とした。また、使用する应用コンテキストに含まれる ASE モジュール間の関係付けを行うことで、SAO のインスタンスを管理するようにした。

b) プリミティブのルーティング制御

特定の应用コンテキストに依存しないルーティングを行うため、

ルーティングの情報をファイルから取り込んで動作し、ルーティングすべき ASE モジュールを間接呼び出しするようにするとともに、プリミティブのデータをポインタ渡しとすることで高速化を図った。すなわち、独自に規定した ASE 識別子を用いて SACF にリンクされる ASE や ASE 間の依存関係を数字で表現してファイルに格納し、SACF はこれを起動時に読み出す。プリミティブの受信時には、図 7 のアルゴリズムに従ってルーティング先を判断し、ASE モジュールの関数へのポインタを用いて対応する ASE モジュールを呼び出し、プリミティブへのポインタを渡すようにした¹⁵⁾。



図 5 ログレコードのファイルフォーマット
Fig. 5 File format for log record.

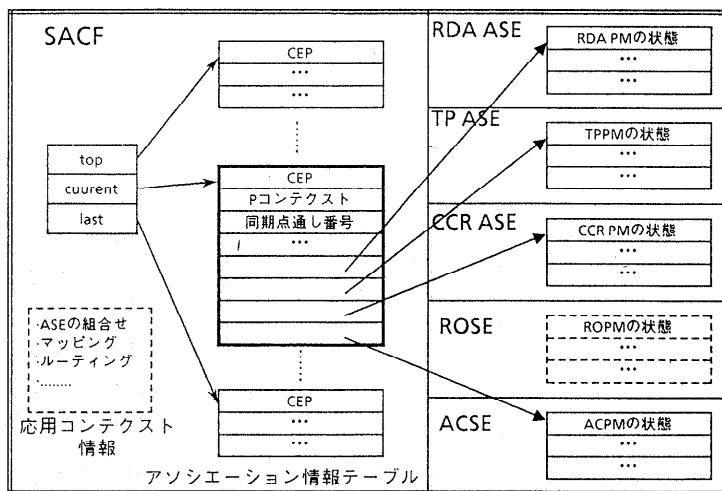


図 6 SACF におけるアソシエーション管理
Fig. 6 Association management in SACF.

具体的には、図8(a)のように、応用コンテキスト名(ap_context_name)、使用されるASE(linked_ase)、ASEの依存関係(upper_module)を表現した。例えばC-BEGINind プリミティブを受信した場合CCRのASE識別子は2であり、図8(b)のようにCCRの上位モジュールを検索するとASE識別子3のPT ASEとなるため、TP ASE モジュールのアドレスを用いて間接呼び出しを行う。また、例えばC-PREPAREindのように、標準では、利用者データにTP APDU が埋め込まれているかどうかで、同一のプリミティブでもルーティング先が異なる場合が存在する。これについては、SACFの処理の簡素化を図るため、ファイルに格納したルーティング情報に従うようにした。

c) 異なる ASE の APDU に対する連結・分離

分離は必須の機能であるが連結は任意であるため、TPSUが連結の開始と終了を指示することとした。ただし、TP-BEGIN-DIALOGUE-RC+UASE-APDUのように自発的に連結できるパターンに関しては、MACFが連結の開始と終了を指示するようにした。

連結の場合、各ASEモジュールは通常のマッピング規則に従って、APDUを作成しプレゼンテーションプリミティブを発行する。SACFは、各ASEから連結する全プリミティブを受信すると、APDUの連結を行うと共に、連結時のマッピング規則を対応させ、必要なら新たなプレゼンテーションプリミティブを作成し発行するようにした。分離の場合には、逆にSACFでAPDUを分離して各ASEに対するプリミティブを作成し、連結されていた順番で各ASEモジュールに渡すようにした。

(2) TP ASE

TP ASEは、MACFからのNNプリミティブや下位からのCCR、ACSE、プレゼンテーションのプリミティブを受信して、TP APDUの作成・解析とサービスのマッピングを行う。この時、同一のNNプリミティブであっても、プレゼンテーション層にマッピン

```

loop:
Receive primitive
if it is Application Layer primitive
then
    if service type of the primitive is either request or response
    then pass the primitive to the ASE module corresponding to ASE identifier
    else pass the primitive to the upper module
    end if
else if it is Presentation Layer primitive
    if service type of the primitive is either request or response
    then pass the primitive to the presentation module
    else pass the primitive to the module corresponding to Presentation Context identifier
    end if
end if
goto loop

```

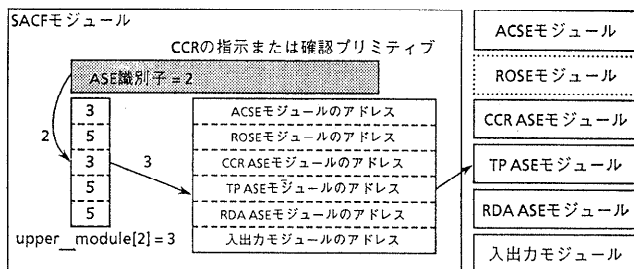
図7 SACFのルーティング制御アルゴリズム
Fig. 7 Routing control algorithm for SACF.

```

struct application_context{
char ap_context_name[]="1-0-9579-2-3-1";
/* RDA-SQL-TP応用コンテキスト-Ver.1 */
int linked_ase[5]={1,0,1,1,1};
/* ACSE(0)-CCR ASE(2)-TP ASE(3)-RDA ASE(4) are linked to SACF */
int upper_module[5]={3,5,3,5,5};
/* ACSE(0)->TP ASE(3) */
/* ROSE(1)->MACF(5) */
/* CCR ASE(2)->TP ASE(3) */
/* TP ASE(3)->MACF(5) */
/* RDA ASE(4)->MACF(5) */
};

```

(a) ASE間の依存関係の表現
(a) Representation of dependency among ASEs.



(b) ASE間の依存関係の探索
(b) Search for dependency among ASEs.

図8 SACFにおけるASE間の依存関係
Fig. 8 Dependency among ASEs in SACF.

グされる場合とCCRにマッピングされる場合がある。プロトコル仕様ではMACFとSAO間でマッピング情報に関するグローバル変数を用いているが、ここではマッピング情報に関するパラメータをNNプリミティブに独自に追加することとした。また、ダイアログの割り当てられていないアソシエーションに残ったAPDU (STRAY APDU) のパーキング機能を持たせ、そのためにダイアログ確立中などアソシエーションの状態を独自に規定した。

5. 評価および考察

5.1 TP プロトコルソフトウェアの汎用性について

前述した資源管理方法とソフトウェア構成法に従い、U-ASEとしてRDA ASEとUDT-ASEを、TPSUとしてデータベースアクセスとEDIメッセージ転送を実現し、汎用性を検証した。

(1) 資源管理における汎用性

データベースアクセスの場合、制御対象データをテーブルとした。TPSUでは、RDAおよびTPのプリミティブを、市販のDBMS(データベース管理システム)であるORACLEのコマンドに対応付けることで資源管理を実現した¹⁶⁾。また、EDIメッセージ転送の場合、TPSUで送受するEDIメッセージをファイルに格納し、制御対象データとした。ファイルの制御は、C言語の標準ライブラリの持つファイルI/Oで実現した。ただし、通常のファイルシステムにはコミットメント機能が存在しないので、一時ファイルの作成により対処した。

その結果、データベースアクセスではDBMSとのインタフェース、EDIメッセージ転送ではファイルシステムとのインタフェースをすべてTPSU内に持たせることができ、TPSPには資源に依存した処理は不要であり、応用が異なっても扱う資源に依存しないTPプロトコルソフトウェアを実現できることが実証できたといえる。この手法は、RPCやネットワーク管理など他の応用にも適用できると考えられる。

(2) ソフトウェア構成における汎用性

RDA ASEの場合は、TPを使用しない基本コンテキストとTPを使用するTPコンテキスト(提供者支援トランザクション)を、UDT-ASEの場合には提供者支援トランザクションの3種類の応用コンテキストを実現した。RDAの2種類の応用コンテキストに関しては、応用コンテキストデータファイルの変更だけで対応できた。また、RDA ASEをUDT-ASEのモジュールに交換したが、SACFとASE間のインタフェースを統一したため、SACFモジュールのプログラムへの変更は不必要であった。これにより、各ASEモジュールがプリミティブの授受やアソシエーション情報へのアクセス方法などのSACFとのインタフェースさえ遵守すれば、必要なASEのオブジェクトモジュールをリンクするだけで任意の応用コンテキストを実現できることが実証できた。また、SACF自

体もルーティング情報など応用コンテキストに関する情報をファイルから読み込んで動作するようにしたため、応用コンテキストに依存しない汎用的なSACFが実現できた。

さらに、特定MACFとしてRDA MACFモジュールを設けることで、基本コンテキストでは共通MACFのアソシエーションプールを利用したり、TPコンテキストではRDA MACFにRDAとTPのプリミティブが混在する状態を管理させることができ、共通MACFと特定MACFのモジュール化の有効性が実証できた。

このように、3章で検討した資源管理方法とソフトウェア構成法により汎用性が高いソフトウェアを実現できたと考えられる。

5.2 性能について

(1) 測定環境

ソフトウェアの実用性を評価するため、TPSUにおける処理の代表的な例としてデータベースアクセスに関する時間を測定した。本測定では、VAX 8700上にルートと2個のリーフの3個のノードからなるダイアログツリーを構成し、各ノードごとにセッション層以上のプロセス群(セッション層:全機能単位、プレゼンテーション層:カーネル機能単位)を用意して、トランスポート層で折り返す形態をとった。応用コンテキストとして、RDAの基本コンテキストとTPコンテキスト(連鎖・非連鎖)を実現した。TPSUとしてRDAのクライアントとサーバを実現し、サーバ側のDBMSにはORACLEを使用した。測定に使用したシーケンスを図9に示す。

(2) 評価結果

表2に、図9のシーケンスが開始されてから完了するまでの時間を測定した結果を示す。この結果には、資源のオープンやコミットなど、TPSUがDBMSにアクセスする時間も含んでいる。

基本コンテキストとTPコンテキストを比較すると、全体的にTPコンテキストの方が処理時間がかかっているが、特にダイアログ開始/資源オープンとコミットする場合に差が大きい。

ダイアログ開始/資源オープンについては、資源オープンの処理は同じであるため、ダイアログ開始時間に差があることになる。これは、TPプロトコルモジュールの内部で使用する変数やテーブル等の数が多いため、その初期化処理に時間がかかるためと考えられる。これについては、アソシエーションプールを利

用することで時間を短縮可能である。実際にアソシエーションプールを利用する場合としない場合の差を測定したところ、約1秒短縮できることを確認した。また、リーフの数が1個と2個の場合で2倍近い時間がかかっているのは、TFSU の処理がシーケンシャルに行われるためであり、この部分を非同期にすることにより時間を短縮することが可能である。

コミット処理で基本コンテキストとTPコンテキストの差が大きいのは、基本コンテキスト

表 2 測定結果 (単位: 秒)

Table 2 Results of performance measurement.

			基本コンテキスト	TPコンテキスト (非連鎖)	TPコンテキスト (連鎖)
(a)ダイアログ開始/ 資源オープン	リーフの 数	1	2.18	4.41	4.96
		2	4.92	6.30	7.28
(b)DB検索*	トランザ クション	中	0.57	0.98	-
		外	0.85	1.08	-
(c)トランザクション 開始	リーフの 数	1	0.11	0.15	-
		2	-	0.20	-
(d)トランザクション 終了(コミット)	リーフの 数	1	0.20	1.53	1.74
		2	-	3.15	3.59
(e)トランザクション 終了(ロールバック)	リーフの 数	1	0.20	0.19	0.32
		2	-	0.31	0.45
(f)資源クローズ/ ダイアログ終了	リーフの 数	1	0.57	0.67	1.82
		2	1.70	1.62	3.94

*:DBの検索は4列4行(1行37バイト)からなるテーブルから1行検索に要した時間

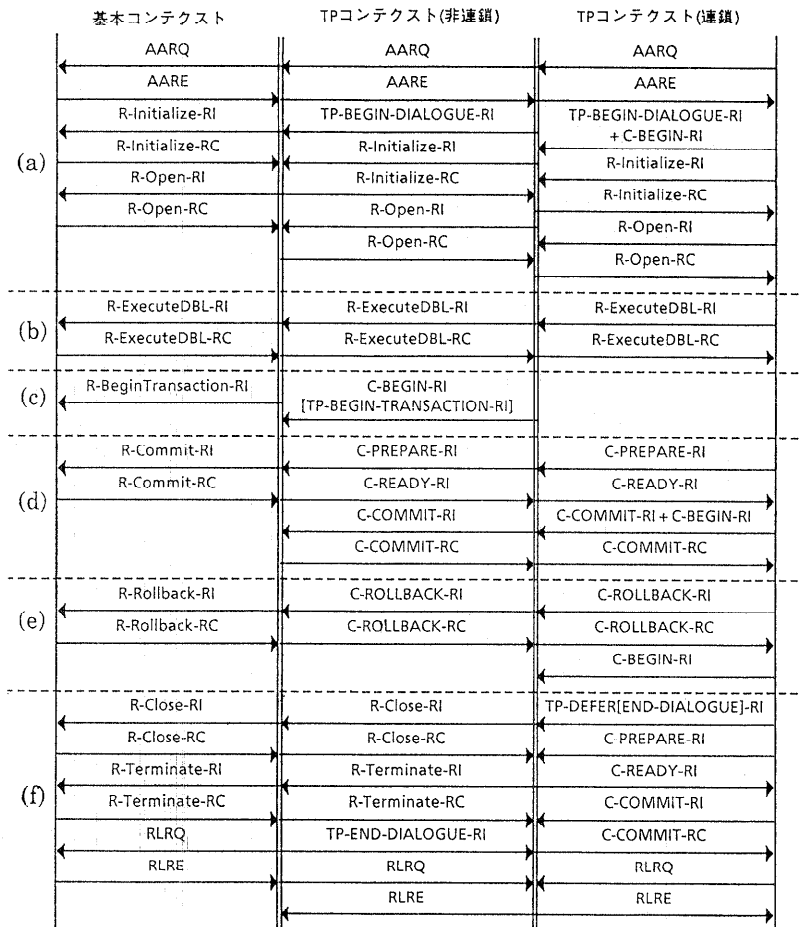


図 9 測定に使用したシーケンス

Fig. 9 Sequence used for performance evaluation.

ではコミット処理が1相であることに加え、ログをとらないことが大きいと考えられる。これはまた、ロールバック処理ではログをとる必要がなく、基本コンテキストとTPコンテキストでロールバック処理の所要時間にはほとんど差がないことから確認できる。ログファイルの書き出し処理時間を別途測定したところ、1ダイアログにつき約150msかかっていることがわかった。この部分の高速化を図るためには、ファイルサイズの縮小やファイルの一元化による書き込み回数の削減を行ったり、ファイル入出力にraw disk I/Oを使用するなどの工夫が必要であると考えられる。

次に、SACFモジュールの導入や共通MACFと特定MACFモジュールの分離など汎用的なソフトウェア構成にしたことによる性能への影響については、ルーティング処理やプリミティブの授受に関しては処理自体は変わらないため、処理時間に与える影響はほとんど無いと言える。ただし、APDUの連結・分離処理でSACFで新たにプリミティブを作成する場合には、多少オーバーヘッドが存在すると思われる。また、TPSUで資源管理を行わせたことについては、性能の観点からは、TPSPにおける通信処理とTPSUにおける資源管理の並行処理が容易になるというメリットがあると考えられる。

以上の結果から、基本コンテキストに比較すると多少処理時間はかかるが、リカバリが可能なことや複数ノードに対するトランザクション処理の実現等を考えると、TPを用いることのメリットは大きいと言える。また、通常会話型処理ではレスポンスタイムの目標値が約3秒であること、実現したプロトコルソフトウェアは基本動作を確認したものであり性能向上のためのチューニングを行っていないこと、ならびに測定したマシンの性能が今日の計算機の性能から考えて6MIPSとそれほど速くなく、かつ同一システム内に3ノード分のプロセスを起動しており負荷が大きいことを考えると、実用的な処理時間で汎用的なソフトウェアを実現できたと考えられる。

6. おわりに

本論文では、各種の応用と組み合わせて利用される汎用的なTPプロトコルソフトウェアの設計手法について論じた。ここでは、応用が扱う様々な資源の種類に対応でき、異なる応用プロトコルと組み合わせて動作させることが容易である汎用性が重要であることを

示し、そのための資源管理の方法とソフトウェア構成について論じた。資源管理については、TPSPがローカル資源を管理する方式とTPSUがローカル資源を管理する方式を比較し、特定の応用が対象とする資源に依存しない後者の方式を採用した。また、ソフトウェア構成については、ALSに基づきMACFモジュールとSACFモジュールを明示的に導入し、ASEモジュールのソフトウェアを部品として柔軟に組み合わせ可能とするために、SACFにはプリミティブのルーティング制御や複数ASE間のAPDUの連結を行わせるようにした。MACFは、MACFを必要とする他の応用プロトコルへ対応するため、各種応用に共通なMACFと応用に特有のMACFモジュールに分離するようにした。さらに、実装したソフトウェアの処理効率を汎用性や処理効率の観点から評価し、実用的レベルの処理効率で汎用的なソフトウェアを実現できることを示した。

謝辞 日頃御指導頂くKDD研究所 小野前所長(現文部省学術情報センター教授)、浦野所長に感謝します。

参考文献

- 1) ISO/IEC DIS10026-1~3, Information Processing Systems—OSI—Distributed Transaction Processing (1989).
- 2) 楠 和浩, 田中功一, 中川路哲男, 勝山光太郎, 水野忠則: OSI トランザクション処理システムの試作, 情報処理学会 DPS 研究会, 45-5 (1990).
- 3) 岩倉伸行, 松田栄之: OSI トランザクション処理プロトコル実装システムの実現方式に関する一考察, 情報処理学会 DPS 研究会, 49-4 (1991).
- 4) ISO/IEC 9545, Information technology—OSI—Application Layer structure (1989).
- 5) 小花貞夫, 杉山敬三, 鈴木健二: OSI におけるトランザクション処理の位置付けに関する一考察, 第36回情報処理学会全国大会論文集, 3G-4, pp. 619-620 (1988).
- 6) 杉山敬三, 小花貞夫, 鈴木健二: OSI トランザクション処理 (TP) プロトコルソフトウェアの設計, 情報処理学会 DPS 研究会, 47-11 (1990).
- 7) X/Open XO/GUIDE/91/020, Distributed Transaction Processing Reference Model (1991).
- 8) 小花貞夫, 西山 智, 杉山敬三, 鈴木健二: OSI における応用層構造 (ALS) の実現方法の提案, 第37回情報処理学会全国大会論文集, 2E-5, pp. 466-467 (1988).
- 9) 小花貞夫, 杉山敬三, 鈴木健二: ソフトウェアモジュールの再利用化を考慮した OSI 応用層ソフトウェアの実装方法, 情報処理学会 DPS 研究会, 55-1 (1992).

- 10) ISO/IEC DIS9804/9805.3, Information Processing Systems—OSI—Commitment, Concurrency and Recovery (1989).
- 11) ISO/IEC DIS9579, Information Processing Systems—OSI—Remote Database Access (1991).
- 12) ISO/IEC CD10026-6, Unstructured Data Transfer (UDT) for TP (1991).
- 13) 小花貞夫, 加藤聰彦, 鈴木健二: OSI プレゼンテーション, ACSE, FTAM プロトコルの実装と評価, 情報処理学会論文誌, Vol. 30, No. 7, pp. 741-753 (1989).
- 14) 杉山敬三, 小花貞夫, 鈴木信雄: OSI CCR における障害回復機能の実現手法, 第 40 回情報処理学会全国大会論文集, 6 N-4, pp. 1458-1459 (1990).
- 15) 杉山敬三, 小花貞夫, 鈴木健二: OSI 応用層プロトコルソフトウェアの部品化のための応用コンテキスト制御方式の提案, 第 44 回情報処理学会全国大会論文集, 4 L-6, pp. 1-157-1-158 (1992).
- 16) 小花貞夫, 堀内浩規, 杉山敬三, 鈴木健二: OSI RDA プロトコルソフトウェアにおけるトランザクション機能の拡充, 第 43 回情報処理学会全国大会論文集, 4 T-8, pp. 1-219-1-220 (1991).

(平成 4 年 6 月 26 日受付)

(平成 5 年 2 月 12 日採録)



杉山 敬三 (正会員)

昭和 37 年生. 昭和 60 年京都大学工学部情報工学科卒業. 昭和 62 年同大学院修士課程修了. 同年国際電信電話(株)入社. 現在, 同社研究所通信網支援ソフトウェアグループ担当主査. この間, OSI プロトコルの実装, ネットワークアーキテクチャ, 分散処理技術, EDI の研究に従事. 電子情報通信学会会員.



小花 貞夫 (正会員)

昭和 28 年生. 昭和 51 年慶応義塾大学工学部電気工学科卒業. 昭和 53 年同大学院修士課程修了. 同年国際電信電話(株)入社. 現在, 同社研究所交換グループ主任研究員. 工学博士. パケット交換方式, ネットワークアーキテクチャ, OSI プロトコルの実装, データベース, 分散処理技術, インテリジェントネットワークの研究に従事. 電子情報通信学会会員.



鈴木 健二 (正会員)

昭和 20 年生. 昭和 44 年早稲田大学理工学部電気通信学科卒業. 昭和 44 年から 45 年までオランダのフィリップス国際工科大学に招待留学. 昭和 51 年早稲田大学大学院博士課程修了. 工学博士. 同年国際電信電話(株)入社. 現在, 同社研究所 OSI 通信グループリーダー. この間, 磁気記録, パケット交換方式, ネットワークアーキテクチャ, 分散処理の研究に従事. 昭和 62 年度前島賞受賞. 平成 5 年度より電気通信大学大学院情報システム学研究科客員教授. 電子情報通信学会, IEEE 各会員.