７Ｖ−５　ＯＳＩ−ＴＰの実装機能範囲に関する一考察

岩倉　伸行　　松田　栄之

ＮＴＴデータ通信株式会社

1．はじめに

国際標準化機構（ＩＳＯ）で規定されている分散トランザクション処理プロトコル（ＯＳＩ-ＴＰ）[1]は、種々のアプリケーションに適用可能な応用層のプロトコルとして期待されている。

このＯＳＩ-ＴＰの実現には、同じ応用層のプロトコルを実現するアプリケーション・サービス要素（ＡＳＥ）を複数必要とする特徴があり、応用層構造（ＡＬＳ）[2]に準拠したプログラムモジュールを用いることで、開発効率やシステムの拡張性を高めることができる[4,5,6]。

一方、ＯＳＩ-ＴＰを利用したシステムは、ＯＳＩ環境における相互接続性の向上を目指すとともに、アプリケーションに応じた厳しい性能要件を満たす必要がある。

そこで、本稿では、ＡＬＳに準拠してＯＳＩ-ＴＰを実装したシステムの開発に関し、アプリケーションに応じた実装機能を選択することによる相互接続性の維持とプロトコル処理性能について検討する。

2．ＯＳＩ-ＴＰの特徴

ＯＳＩ-ＴＰは、機能単位の考え方を採用しており、コミットメント[3]の実現や通信制御権の移動等の機能を選択するようになっている。システム開発時には、これらの機能選択をする必要があり、例えば、文献[7]では、６つの機能選択パターンに分類して、実装システムの相互接続性を高めようとしている。

さらにＯＳＩ-ＴＰを利用したデータの転送は、利用者ＡＳＥ（Ｕ−ＡＳＥ）に抽象化されており、非構造型データ転送[7]の様な簡単なものから、セションのトークンの移動や同期点の設定を行う必要のあるものまで実装可能になっている。

3．実装上の検討項目

次の観点から実装範囲について検討する。

[実装の観点]
（１）プロトコル処理効率に重点をおく観点
（２）システムの拡張性に重点をおく観点

（１）の観点からは、実装時には対象範囲外の機能単位をなくし、実装範囲をできるだけ絞り込み冗長な開発を行わない実装仕様を決めることが望ましい。これに対し（２）の観点からは、将来のプロトコル実装範囲の拡張を考慮した実装仕様を規定することになり、両者は一般に相反する。そこで実装機能の選択と開発規模について検討する。

3．1　ＯＳＩ-ＴＰの実装で選択する機能

（１）機能単位の選択

本稿では、文献[7]より代表的な２つのプロフィルを取り上げ、状態遷移表のセル数（表１）の観点で比較する（表２）。

①全二重型制御、応用トランザクション（ＡＰ３１１）
②全二重型制御、非連鎖トランザクション（ＡＰ３１２）

表２より両プロフィルの比較結果から、ＯＳＩ-ＴＰ実装上の中核を占めるＭＡＣＦの状態遷移表規模に大きな差があることが分かり、機能単位の選択による実装規模に大きな影響を及ぼすことが分かる。

表１　状態遷移表のセル数

モジュール名	機能単位	セルの概数
ＭＡＣＦ	ダイアログ	２００
	ハンドシェイク	１００
	コミットメント	１６０
	ロールバック	１２０
	チャネル	４０
ＳＡＣＦ		１７０

（２）アソシエーション・プール機能の選択

ＯＳＩ-ＴＰではアソシエーションプールの機能はオプショナルとして実装されるものである。この特徴を表３にまとめる。これより機能選択による実装アルゴリズムの複雑さが異なることが分かり、プロトコル規定に現れない機能の実装に関しても大きな影響を及ぼすことが分かる。

（３）障害回復機能の選択

提供者支援トランザクションを実現する場合は、ＯＳＩ-ＴＰが障害回復機能を実現する必要がある。プロトコル規定で障害回復の範囲と手順が規定されており、このことから、機能単位の選択

表２　プロフィルと開発規模

プロトコル ＼ プロフィル	応用トランザクション（ＡＰ３１１）	提供者支援トランザクション（ＡＰ３１２）
セションプロトコル（機能単位）	カーネル 全二重	カーネル 全二重 制御データ 小同期 大同期 再同期
応用層のプロトコル（ＡＳＥ種別）	ＴＰ−ＡＳＥ Ｕ−ＡＳＥ ＡＣＳＥ	ＴＰ−ＡＳＥ Ｕ−ＡＳＥ ＡＣＳＥ ＣＣＲ−ＡＳＥ
ＯＳＩ-ＴＰ（ＭＡＣＦの状態遷移表のセル数）	ダイアログ（２００）	ダイアログ（２００） コミットメント（１６０） ロールバック（１２０）

表３　アソシエーションプールと実装機能

	プール機能を実装		プールを実装しない（ＡＰ３１１、ＡＰ３１２）
	応用トランザクション（ＡＰ３１１）	提供者支援トランザクション（ＡＰ３１２）	
応用層のプロトコル	ＴＰ−ＡＳＥ Ｕ−ＡＳＥ ＡＣＳＥ	ＴＰ−ＡＳＥ Ｕ−ＡＳＥ ＡＣＳＥ ＣＣＲ−ＡＳＥ	ＴＰ−ＡＳＥ Ｕ−ＡＳＥ ＡＣＳＥ （ＣＣＲ−ＡＳＥ）
再利用の条件	①アプリケーションコンテキストの一致 ②プレゼンテーションコンテキストの一致	①アプリケーションコンテキストの一致 ②プレゼンテーションコンテキストの一致 ③C-BEGIN-FEAR状態の解消	利用しない
その他	アソシエーションの属性（ＣＷ／ＣＬ）、ＢＩＤの利用条件等により、アプリケーションが必要とするアソシエーションを選択する可能性がある		アソシエーションの利用終了後、管理をする必要はない
実装内容	通常のアソシエーション管理の他に、「再利用条件」、「その他」の条件を考慮したプログラム開発が必要		通常のアソシエーション管理。

A Study on Implementation of OSI distributed Transaction Processing protocol
Nobuyuki IWAKURA, Shigeyuki MATSUDA
NTT DATA COMMUINCATIONS SYSTEMS CORPORATION

表4　障害回復機能

	応用トランザクション（AP311）	提供者支援トランザクション（AP312）
規定範囲	特になし（MACFでは規定されていない）	TPチャネルの実現（MACFセル数：40）
規定範囲外	アプリケーションの必要に応じて ①アソシエーション ②ダイアログ の再確立処理が必要	アプリケーションの必要に応じて ①アソシエーション ②ダイアログ ③トランザクション の再確立処理が必要

による実装時のアルゴリズムへの影響が大きいことが分かる（表4）。さらにシステムの実用性を考えると、ＯＳＩ-ＴＰの規定外の事項であるダイアログの回復処理等についても検討する必要があり、さらに実装規模が大きくなることになる。

3．2　ＳＡＣＦの実装機能への影響

ＡＬＳ準拠のモジュール構成を実現するため、文献[1]では明確化されていないＳＡＣＦの機能を実装時に決める必要がある。具体的には、アソシエーション上でＡＳＥとＳＡＣＦが必要とする情報の管理範囲を決定しなければならない。ここでは、ＳＡＣＦで管理する必要のある情報の範囲が、ＴＰとＡＳＥ（U-ASE）で実装を選択した機能に影響を受けることを、ＡＳＥ間の共通情報であるセションのトークン／同期点を例に、表5にまとめる。

このことから、ＳＡＣＦ内の処理の効率化あるいはシステムの拡張性を選択するために、ＡＳＥの利用機能範囲を明確化することが重要な要素であることが分かる。

4．機能選択の実現方法

通信プログラムの開発では、ソースコードの開発規模と同様に実行モジュール作成時の規模も資源利用などの観点から重要である。そこで実行モジュール作成方式として考えられる次の３つの機能選択方法について比較する。

［方式１］

プロフィルに応じたもののみをソースコードから開発する。ただし、ＡＬＳに準拠することでＡＳＥを新たに開発する必要はなく、ＡＳＥの組み合わせとＳＡＣＦインタフェース部分に対する変更を行う。プロトコル拡張時に必要となる工程稼働を考えた場合、初期の開発に比べ修正規模が大きくなる可能性があるため、機能拡張には向いていない。しかし開発されたものの実行モジュールは小型化され、冗長な処理は含まずに効率的なプロトコル処理が実現される。

［方式２］

ソースコードではプロフィル選択外の開発も行い、アプリケーションに必要な機能に応じて選択コンパイルを行う。

コンパイル時の分岐条件が多くなる可能性があり、プログラム構成上は複雑になるが、種々のアプリケーションに応じたシステム構築が可能になる。初期の開発規模は大きくなるが、プロトコル拡張時の工程稼働は少なくなる。また実行モジュールでも小型化が可能である。

表5　ＳＡＣＦで実装する機能

プロフィル / U-ASEの機能	応用トランザクション（AP311）	提供者支援トランザクション（AP312）
セション の機能を利用しない （トークン、同期点）	・管理情報はない	○トークン（大同期、小同期、データ） ○同期点の管理
セション の機能を利用する （トークン、同期点）	○トークン、同期点 （U-ASEで利用する範囲を管理） ・複数のU-ASEが存在する場合は調整が必要	○トークン（大同期、小同期、データ） ○同期点の管理 ・U-ASEとCCR-ASEの利用で調整が必要 ・複数のU-ASEが存在する場合は調整が必要

表6　ＡＰ３１１の開発対象セル数

開発方式 / モジュール	ソースコードレベルでの開発セル数	実行モジュールレベルでのセル数
方式1 （AP311の範囲を開発）	370	370
方式2 （AP311を選択コンパイル）	790	370
方式3 （AP311をSGで選択）	790	790

［方式３］

ソースコードではプロフィル選択外の開発も行い、アプリケーションに応じてプログラム・ヘッダ・ファイルを変更して種々のアプリケーションに対応する（ＳＧ）。

選択コンパイルを利用するより複雑さは少ないが、実行モジュールは大きくなる。

３つの開発方式をＡＰ３１１に適用した場合、ソースコードと実行モジュールの規模を示す目安として状態遷移表の開発対象となるセル数を比較したものを表6に示す。

5．考察

（1）ＯＳＩ-ＴＰを利用したアプリケーションが限定されているシステムでは、開発工程の観点から冗長な機能単位を実装しないように注意しなければならない。例えば、アソシエーションプールの場合では、機能単位の選択により管理の必要な範囲が広がり、開発規模が大きくなってしまうことになる。

（2）将来のアプリケーション機能の拡張、接続システムの拡張の可能性が高い場合、機能は全て実装し、現在のアプリケーションの要望に応じて選択コンパイルやＳＧを行う方法により実現できる。

（3）ＡＳＥの実現では、プロトコル検証の機能や管理機能の実現を選択できるようにしておくことにより、処理アルゴリズムを最適化できる。特にＡＬＳではモジュール分割されているため、プロトコルチェック機能実現を選択できることがプログラム開発と実用化の過程で有効である。

6．まとめ

本稿では、ＯＳＩ-ＴＰを実装システムの開発に関し、実装機能を選択することによる相互接続性の維持とプロトコル処理性能について、代表的なプロフィルを例として用いた場合を検討した。

ＯＳＩ-ＴＰは、多様なアプリケーションに適用可能なプロトコルとして汎用的に規定されているため、機能単位の選択によりシステム開発の生産性に大きな影響を与える。さらに機能単位の選択は、アソシエーションプール管理のように、ＯＳＩ-ＴＰの規定範囲外の機能にも及ぼす影響が大きいことから、接続するシステムの実装機能や将来の拡張予定に応じたプロトコル処理機能を選択し、実装方式を検討する必要があることを示した。

今後は、アプリケーションプログラムの処理特性を含めたシステム性能を考慮して、実装方式の検討を行う予定である。

［参考文献］

[1]ISO/IEC DIS 10026 Distributed Transaction Processing, (1991).
[2]ISO/IEC 9545 Application Layer structure, (1989).
[3]ISO/IEC 9804/9805 Commitment, Concurrency and Recovery service element, (1990).
[4]松田、澤、岩倉：「ＯＳＩ-ＴＰ実装方式に関する一考察」、情処全大、第４２回、(1991).
[5]岩倉、松田：「ＯＳＩトランザクション処理プロトコルの実装方式に関する一考察」、マルチメディア通信と分散処理研究会、(1992).
[6]岩倉、澤、玉置：「ＯＳＩトランザクション処理プロトコル実装システムの試作」、情処全大、第３９回、(1989).
[7]JIS X 5003-1987参考「分散トランザクション処理実装規約」、日本規格協会、(1990).