動的ネットワーキングアーキテクチャにおける応用の一例: アプリケーション指向フロー制御システムの提案

北形 元† 菅沼 拓夫† 木下 哲男* 白鳥 則郎†
†東北大学電気通信研究所
E-mail: {minatsu,suganuma,norio}@shiratori.riec.tohoku.ac.jp
*東北大学情報シナジーセンター
E-mail: kino@shiratori.riec.tohoku.ac.jp

本研究では，動的ネットワーキングアーキテクチャにおけるQoS制御機能とネットワーク運用機能に焦点をあて，同機能を利用した応用の一例として，アプリケーション指向フロー制御システムを提案する．本システムは，アプリケーションの状態を監視し，これに応じてアプリケーションのフロー毎の優先度を決定し，流量を調整するシステムであり，次のような特徴を持つ．(1)論理ネットワーク層の上位に位置するやわらかいネットワーク層内で流量調整を行うため，IPルータ等への特別な実装を必要とせず，利用するネットワーク環境が限定されない．(2)やわらかいネットワーク層からアプリケーションの状態を監視し，フローの優先度を決定するため，既存アプリケーションに特別な実装が不要．本論文では，提案システムの設計および実装について議論し，実験を通じ提案システムの有効性を確認する．

An Application of Dynamic Networking Archtecture: Application Oriented Flow Control System

Gen Kitagata†, Takuo Suganuma†, Tetsuo Kinoshita* and Norio Shiratori†
†Research Institute of Electrical Communication, Tohoku University
E-mail: {minatsu,suganuma,norio}@shiratori.riec.tohoku.ac.jp
*Information Synergy Center, Tohoku University
E-mail: kino@shiratori.riec.tohoku.ac.jp

In this research, we focus on two units, which are QoS Control Unit and Network Management Unit, on Dynamic Networking Architecture. For example of its application, we propose Application Oriented Flow Control System. The system monitors applications' states, decides priority of each application's network flow and controls the flow. The system has two remarkable characteristics. (1)Because the system controls network flow in the Flexible Network Layer which lays on the logical network layer, it requires no special implementation of network environments. (2)Because the system decides priority of each application's flow by monitoring the applications, it requires no special implementation of applications. In this paper, we discus about design and implementation of this system.

1 はじめに

近年，ISDNやADSLなどのデジタル回線の普及に伴い，自宅等からのインターネット利用者が急増している．このような，上流のネットワークに比べ加入者回線の方が低速な環境においては，同時に利用しているアプリケーションのトラフィックにより，他のアプリケーションのトラフィックが圧迫され，サービス品質が低下する場合がある．具体的には，FTPによるファイル転送中に生じるTelnetのキャラクタエコー遅延等のレスポンス低下や，ビデオストリーミングのコマ落ち等の品質低下があげられる．このため，アプリケーションのフロー単位での流量調整等，きめの細かいネットワークサービス品質(以降QoSと略記)保証の実現が重要な課題となっている．

ネットワークQoS保証に関する代表的な方式として，IETFによりIntServ[1]やDiffServ[2]が提案されている．IntServはエンド・ホスト間のQoS保証を行う仕組を提供し，DiffServはサービスクラスごとの相対的な優先制御を行うことにより，IntServ利用時のスケーラビリティの問題を解決している．しかし，両方式とも経路上のIPルータにQoS保証のための特別な実装を必要とし，更に，これらを利用するアプリケーション側にも両方式を利用するための実装が必要となる．このため，利用できるネットワーク環境やアプリケーションは限定されており，上記方式に対応してない有用な既存アプリケーションの利用者の多くが，これらのネットワークQoS保証方式を利用できないという問題がある．

我々は，上述の問題を含む，アプリケーションと論理ネットワーク間のギャップにより生じる種々の問題を解決するため，動的ネットワーキングアーキテクチャを提案している[3, 4]．動的ネットワーキングアーキテクチャの特徴は，利用者／アプリケーションレベルでの変動や，基盤となる現行のIP ネットワークなどの論理ネットワークレベルでの変動を自律的に監視・調整・制御する「やわらかいネットワーク層」と呼ぶ機構を新たに導入する点にある．本研究では，やわらかいネットワーク層が持つ機能群のうち，QoS制御機能とネットワーク運用機能に焦点をあて，同機能を利用した応用の一例として，上述のネットワークQoS保証方式の問題を解決するアプリケーション指向フロー制御システムを提案する．本システムの特徴は，(1)論理ネットワーク層の上位に位置するやわらかいネットワーク層内で流量調整を行うため，IPルータ等への特別な実装を必要とせず，利用するネットワーク環境が限定されない点，(2)やわらかいネットワーク層からアプリケーションの状態を監視し，フローの優先度を決定するため，既存アプリケーションに特別な実装をせずに利用可能である点である．本論文では，提案システムの設計および実装について議論し，実験を通じ提案システムの有効性を確認する．

2 動的ネットワーキングアーキテクチャ

従来のネットワークアプリケーションの多くは，TCP/UDPなどが提供するトランスポート通信サービスを静的/固定的に用いて構成されてきた．しかしながら，時々刻々と変化する利用者要求やネットワーク環境の中で，従来の固定的なアーキテクチャの弊害として，様々な問題が生じつつある．例えば，通信相手との通信路確保のために，通信相手に対応するIPアドレス，ポート番号を推定しなければならないといった接続時の不自由さや，アプリケーション毎・セッション毎のQoS制御が柔軟に行えないなどの問題がある．これらの問題は，様々な通信基盤上で頻繁に移動しながら通信を行う将来のネットワーク利用形態を考えると，今後更に大きな問題となるであろう．これらの問題を解決するためには，ソフトウェア工学的，あるいは利用者指向的な観点を考慮した新たなネットワークアーキテクチャを構成する必要がある．

そこで我々は，これらの問題点を解決するために，これまで行ってきたやわらかいネットワークに関する研究成果に基づき，次世代広域ネットワーク環境のモデルとして，動的ネットワーキングアーキテクチャを新たに提案している．

図1に本アーキテクチャの機能構成を示す．現行のIPネットワーク等のサービス機能層(論理ネットワーク層:LN層)とアプリケーション層(AP層) の中間に位置する機能層として，新たに「やわらかいネットワーク層(FN層)」を導入する．FN層は，AP層からの要求とLN層のサービスを効果的に連結し，AP 層とLN層の双方における大・小/一時的・長期的変動を自律的に調整する機能を持つ．以下に，図1中の各機能の概要を述べる．

(1) メディアエレメント群

AP層にて利用される様々なメディア(動画，静止画，音声等)を提供するコンポーネント群，およびcodecコンポーネント群等を管理する．具体的には，利用者要求，LN層の状況に応じて最適なコンポーネント群を選択，生成し，また動作状況に応じてそれらの再構成を行う．またレガシーアプリケーションとの連携を考慮し，レガシーアダプタ機能を提供する．AP層では，これらのメディアエレメントを利用することで，アプリケーションの通信にかかわる部分を構築する．

(2) 高レベル通信機能

- 位置変化獲得・名前解決機能: AP層のエンティティおよび利用者の位置を把握する．またこの情報を基に，IPアドレスとポート番号よりもよりAP/利用者に近い表現形式(人物名やサービス要求仕様)で，接続相手を特定する．
- 接続要求獲得機能: AP層からの接続要求を管理する．
- 接続提供機能: メディアエレメント間の通信路生成・管理を行う．

(3)QoS制御機能

- QoS要求獲得機能: 上位層からのQoS要求を獲得する．
- NW(ネットワーク)情報収集機能: LN層あるいは上位層から，ネットワークにかかわる様々な情報を収集，管理する．

(4) ネットワーク運用機能

- 流量調整機能: 後述する流量解決機能と連携して，アプリケーションのフロー毎の流量を調整する．
- 経路多重化機能: 後述する経路解決機能と連携して，LN層が提供する通信路の切替え，多重化等を行う．
- 流量解決機能: メディア単位での流量を決定する

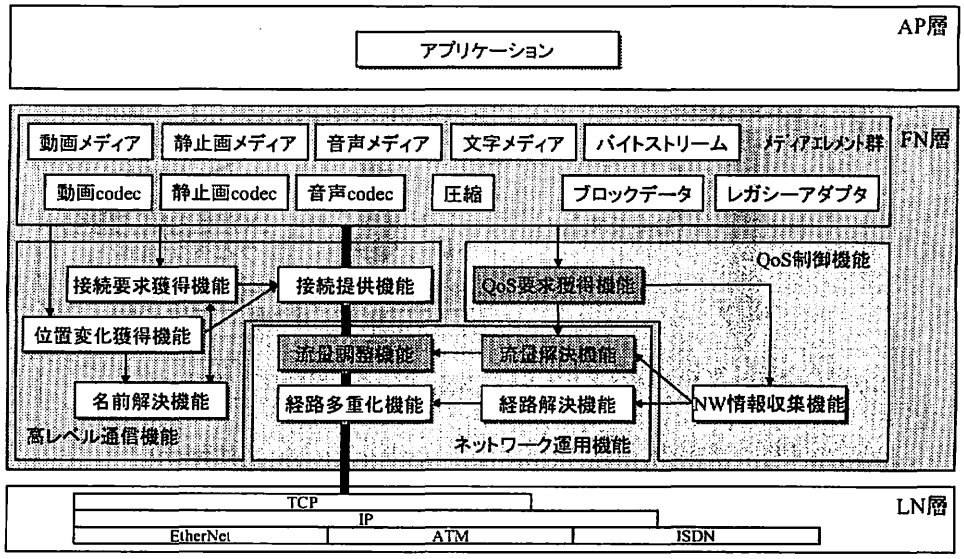

図 1: 動的ネットワーキングアーキテクチャの機能構成

- 経路解決機能: LN層の提供する通信路の利用法を決定する.

以上のFN層内の各機能は, AP層からの要求, LN層の状態, FN層自身の動作状況などに応じて, 動的に構成される. また, 必要に応じてFN機能要素を再構成し, FN層自体が適応的に振舞う.

3 アプリケーション指向フロー制御システムの提案

3.1 既存のQoS保証方式の問題点

代表的なQoS保証方式であるIntServやDiffServは, 経路上のIPルータにRSVP[5]等のシグナリングプロトコルの実装や, DSフィールド扱うための実装を要求する. このため, アプリケーションが要求する帯域や最大遅延を保証できるという利点の一方で, (P1)利用できるネットワークが限定されるという問題を含んでいる. また, IntServやDiffServを利用するアプリケーションには, シグナリングプロトコルを扱ったり, DSフィールドにDSCPをセットするための実装が不可欠である. このため, 既存アプリケーションを両方式に対応させるためには(P2)既存アプリケーションの変更が必要という, アプリケーションの再利用性の観点からみた問題点がある.

3.2 提案システム

本論文では, 上述の2つの問題(P1)(P2)を解決するQoS保証方式として, FN層の3つの機能, QoS要求獲得機能, 流量解決機能, 流量調整機能を利用したアプリケーション指向フロー制御システムを提案する.

提案システムではまず, IntServ等により論理ネットワーク層レベルで行われていたフロー制御を, FN層, すなわち従来のアプリケーション層レベルで行うことで, 前述した(P1)利用できるネットワークが限定されるという問題を解決する. 具体的には, アプリケーション層と論理ネットワーク層間に流れるトラフィックを, 流量調整機能によりフロー毎にキューイングし, それぞれのキューからデータを取り出す頻度を優先度に応じ制御する. これより, 論理ネットワーク層のIPルータによるフロー制御機能に依存することなく, 様々なネットワーク環境においてフロー制御が可能となる.

加えて, QoS要求獲得機能とQoS解決機能により, 利用者が操作しているアプリケーションのウィンドウ状態に応じ各アプリケーションのフローの優先度を決定することで, 前述の(P2)既存アプリケーションの変更が必要という問題を解決する. 具体的には, QoS要求機能がアプリケーションのウィンドウの重なり順序や大きさ等のウィンドウ状態を監視し, これに基づきQoS解決機能がアプリケーション間の優先順位を決定する. これにより, 既存のアプリケーションに特別な変更を施すことなく, 利用者がアプリケーション毎のフロー優先度を調整することが可能となる.

図2に, 従来のIntServ,DiffServにおけるQoS制御機能モデルと提案システムにおけるQoS制御機能モデルの比較を示す. 従来手法に対する提案システムの特徴は, QoS制御に必要となる機能をFN層に配置することにより, QoS保証に非対応なアプリケーション, 及びQoS非保証型のネットワークにおいてもフロー制御機能が利用

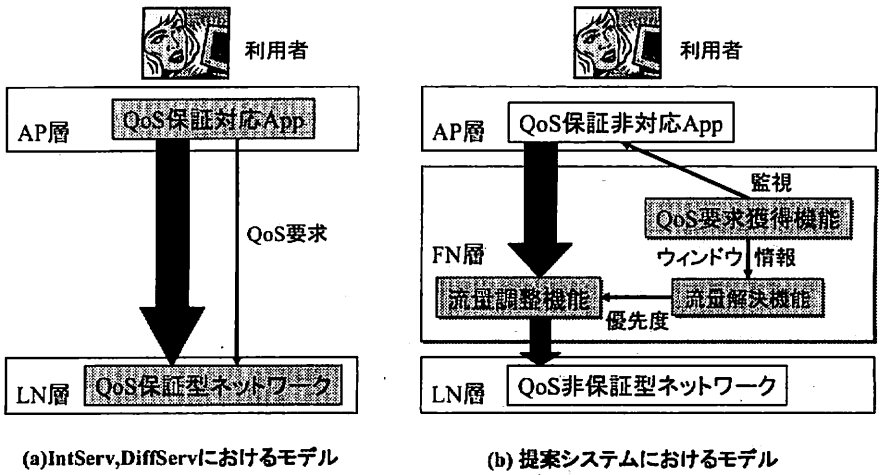

図 2: QoS 制御機能モデルの比較

可能である点である．

4 設計と実装

本章では，前述の提案システムを構成する以下の３つの機能の詳細設計，及び実装について述べる．

- QoS 要求獲得機能
- 流量解決機能
- 流量調整機能

4.1 QoS 要求獲得機能の設計

QoS 要求獲得機能では，利用者がどのアプリケーションのフローを優先したいのかを把握するために，アプリケーションのウィンドウ情報を取得する．ウィンドウ情報としては具体的に

- ウィンドウ識別子
- アプリケーション識別子
- ウィンドウの z 座標（重なり）
- ウィンドウサイズ
- 最大化・最小化

などの情報を取得する．取得したウィンドウ情報は，流量解決機能に伝達される．

4.2 流量解決機能の設計

流量解決機能では，QoS 要求獲得機能から得られたウィンドウ情報からQoSパラメータを生成し，流量調整機能に伝達する．ウィンドウ情報とQoSパラメータは，それぞれ表1，2のように表現する．流量解決機能には，ウィンドウ情報を条件部，QoSパラメータを動作部としてもつIF-THEN ルールを，流量解決知識として組み込む．この流量解決知識とQoS 要求獲得機能から得られたウィンドウ情報に基づいて推論を行い，実際のQoSパラメータを生成する．なお，生成するQoSパラメータは，アプリケーション間の優先度を比率で表したものである．

ウィンドウ情報	
classname	アプリケーション名
handle	ウィンドウ識別子
activity	ウィンドウアクティブ
zorder	z 座標
max/min	最大化/最小化
width・height	ウィンドウサイズ

表 1: ウィンドウ情報の表現形式

QoS パラメータ	
classname	アプリケーションクラス名
handle	ウィンドウ識別子
ratio	送信率

表 2: QoS パラメータの表現形式

図3にネットワーク情報とウィンドウ情報，及びQoSパラメータ生成ルールの例を示す．

図3のネットワーク情報は，利用者が使用しているネットワークの環境を示す情報である．ネットワーク情報とウィンドウ情報から，ルールで書かれた知識により推論を行い，QoSパラメータが生成される．

```
//ネットワーク情報//
(network_state :width low :delay normal :reliability high)

//ウィンドウ情報//
(window_info :classname IEFrame :handle 8D5 :activity no
             :zorder 2 :max no :mini no :width 223 :height 156)
(window_info :classname VTwin32 :handle D77 :activity yes
             :zorder 1 :max no :mini no :width 303 :height 236)

//ルール//
(window_info :classname ?c :handle ?h :activity yes
             :zorder 1 :max ?max :mini ?mini :width > 100 :height > 100)
(network_state :width low :delay normal :reliability high)
-->
(qos_param :classname ?c :handle ?h :ratio 0.8 ) //QoSパラメータ//
```

図 3: ネットワーク情報とウィンドウ情報，及びQoSパラメータ生成ルールの例

4.3 流量調整機能の設計

流量調整機能では，流量解決機能で生成されたQoSパラメータに基づき実際に通信データの送受信制御，すなわちフロー制御を行う．本機能では，アプリケーションのフロー毎にキューをもうけ，フロー単位の流量制御を行う．また，キューから一度に取り出すデータ量(スライス)を変更可能とすることにより，即応性を優先したい場合はスライスを小さくし，反対にスループットを優先したい場合はスライスを大きくするなど，柔軟なキューイングが行えるよう工夫している．

4.4 実装

前章の設計に基づき，提案システムの実装を行った．OSはMicrosoft Windows Meを用い，実装言語にはVisual C++，Java言語，TAF[6]を用いた．QoS 要求獲得機能におけるウィンドウ情報の取得には，Microsoft WindowsのメッセージフックAPIを利用した．また，流量調整機能については，HTTPとTelnetのプロキシとして実装した．

5 実験と評価

前章の設計に基づき実装した提案システムを用いて，有効性を確認するための実験を行った．

5.1 実験環境

図4に実験環境を示す．Client，GatewayにはPC(CPUはPentium4 1.4GHz，OSはMicrosoft Windows Me)を用いた．Web Server，Telnet ServerにはSun MicrosystemsのUltraSPARCStation(OSはSolaris7)を用いた．ClientはQoS非保証型のネットワーク(9600bps，シリアルPPP接続)によりGatewayに接続し，ClientではアプリケーションとしてWebブラウザとTelnetクライアントを使用した．Webブラウザ，Telnetクライアントには，QoS保証非対応であるInternet Explorer，TeraTermをそれぞれ用いた．ClientとGateway間の通信速度は，Gatewayより上流のネットワークの帯域に比べ十分低速な速度として，9600bpsを用いた．

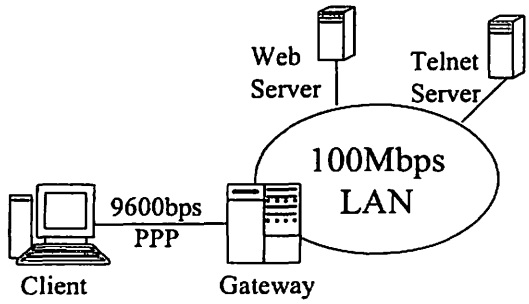

図 4: 実験環境

5.2 実験1：応答速度向上の検証

提案システムの利用による応答速度向上検証のため，HTTPフローが支配的な状況下において，提案システムのQoS制御をOn／Offとしたそれぞれの場合について，Telnetのキャラクタエコーの応答時間を測定した．

実験条件

- HTTPにてファイルをダウンロードしており，常にHTTPフローがPPP回線を占有している
- QoS制御On時のTelnetとHTTPのフロー優先度の比率は1：1
- QoS制御On時に，流量調整機能のキューからデータを取り出すスライスのサイズは1バイト

	応答速度
QoS制御Off時	9.35[sec]
QoS制御On時	約1.00[sec]

表 3: 応答速度向上の検証実験の結果

優先度比率(HTTP:Telnet)	Telnetのスループット
1:2	117.80[bps]
1:20	274.89[bps]

表 4: スループット向上の検証実験の結果

- Telnetのキャラクタエコーは，十分な間隔(1min以上)をあけ1文字ずつ送信し測定する

実験1の結果を表3に示す．なお，実験結果は10回の試行の平均で表している．

実験結果より，QoS制御Offの場合に比べ，QoS制御Onの場合の応答速度が向上していることが確認された．これは，QoS制御On時の流量調整機能の働きにより，TelnetとHTTPのフローが1：1の比率で平等に流れ，Telnetの応答速度を確保できたことを示している．

5.3 実験2:優先度変更に伴うスループット向上の検証

HTTPとTelnetのフロー優先度の比率を1:2から1:20に変化させた場合に，これに応じてTelnetのスループットが向上することを検証する．本実験ではTelnetのスループットを測定するために，Telnetにより一定量のファイルをダウンロードし，ダウンロード完了までの時間を測定した．

実験条件

- HTTPにてファイルをダウンロードしており，常にHTTPフローがPPP回線を占有している
- QoS制御は常にOn
- TelnetとHTTPのフロー優先度の比率は1：2，及び1:20とする
- 流量調整機能のキューからデータを取り出すスライスのサイズは1バイト

実験2の結果を表4に示す．なお，実験結果は10回の試行の平均で表している．

実験結果より，Telnetのフロー優先度を1:2から1:20に上げると，これに伴いTelnetのスループットが向上することが確認された．これは，フロー優先度に応じたアプリケーションのフロー制御が，提案システムの流量調整機能の働きにより行われたことによる．

5.4 評価

実験1，実験2の結果により，QoS非保証型ネットワーク環境において，QoS保証に非対応のアプリケーションを利用する場合でも，提案システムによりアプリケーションごとのフロー制御が可能となり，問題(P1)(P2)が解決されることが検証された．

6 むすび

本論文では，動的ネットワーキングアーキテクチャにおける応用の一例として，アプリケーション指向フロー制御システムを提案した．また，提案システムの設計／実装について述べ，実験を通じ，提案システムの有効性を検証した．

参考文献

[1] J. Wroclawski: The Use of RSVP with IETF Integrated Services, IETF Request for Comments 2210 (1997).

[2] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, W. Weiss: An Architecture for Differentiated Service, IETF Request for Comments 2475 (1998).

[3] T. Kinoshita, S. Sugiura, T. Suganuma, K. Sugawara, and N. Shiratori: Evolutional Flexible Network, *IPSJ Technical Report*, 99(56):143–148, 1999.

[4] T. Suganuma, T. Kinoshita, and N. Shiratori: Flexible network layer in dynamic networking architecture, *Proc of The 1st International Workshop on Flexible Networking and Cooperative Distributed Agents (FNCDA2000)*:473–478, 2000.

[5] R. Braden, Ed., L. Zhang, S. Berson, S. Herzog, S. Jamin: Resource ReSerVation Protocol (RSVP) - Version 1 Functional, IETF Request for Comments 2205 (1997).

[6] 原 英樹, 今野 将, 菅原 研次, 木下 哲男: ソフトウェアエージェント開発教育用システムTAFの設計と実装, 電子情報通信学会論文誌 Vol.J84-D-I No.8, pp.1202–1210 (2001).