推薦論文

分散マルチメディアシステムにおけるアプリケーション QoS 管理手法

加藤由花† 箱崎勝也†

コンピュータネットワーク上に多くの分散マルチメディアシステムが導入されるにつれ，利用者に対して提供されるサービスの品質（Quality of Service: QoS）に対する注目が高まってきている．本稿ではこのサービス品質をアプリケーション QoS と定義し，システム環境に対して適応的なアプリケーション QoS 制御を実現する，アプリケーション QoS 管理システム（application QoS Management System: **QMS**）を提案する．**QMS** はアプリケーション QoS を制御対象とするが，計算機やネットワークの資源が動的，静的に変化する実システム環境に適応するため，複雑なモデル化をともなわないフィードバック型の簡易な制御方法を採用する．また，一般に各利用者や各アプリケーションはそれぞれに要求する QoS レベルが異なり，その QoS 間で交渉が必要であるため，**QMS** に QoS 管理ポリシーの設定機能を実装し，ポリシーに基づいた QoS 交渉が容易に行える仕組みを構築する．我々はさらに，**QMS** によって複数アプリケーション間の QoS 交渉が実現し，システム全体として QoS 管理ポリシーに従ったシステム運用が可能になることを検証するため，シミュレーション実験を行った．本稿ではその結果についても報告する．

Application QoS Management for Distributed Multimedia Systems

Yuka Kato† and Katsuya Hakozaki†

As a large number of distributed multimedia systems are deployed on computer networks, Quality of Service (QoS) for an application becomes more important. This paper defines it as application QoS, and proposes the application QoS Management System (**QMS**). Its control targets are application QoSes and it controls the QoSes according to the system environment. At that time, the system adopts simple control methods without complicated modeling in order to adapt **QMS** to practical system environment. This is because system resources always change dynamically and statically on the practical environment. In addition, we implement the function setting QoS management policy for **QMS**, because each user or each application generally requires a different level of QoS and the system have to negotiate them. This function makes the QoS negotiation easy. By the simulation experiment, we confirmed that the system made it possible to negotiate the QoS between many applications and it was able to manage the whole applications according to the policies. In this experiment, we were able to decide the QoS management policies easily by using **QMS**.

1. はじめに

コンピュータネットワーク上に多くの分散マルチメディアシステムが導入されるにつれ，利用者に対して提供されるサービスの品質（Quality of Service: QoS）に対する注目が高まってきている．データリンク層で通信品質を保証する技術として ATM（Asynchronous Transfer Mode）が実用化されているが，インターネットが広く普及した現在では，エンドシステムまで完全な ATM 化が推進するとは考えにくい．そこで，インターネット上で QoS の制御を行いたいという要求が生じてくる．しかし，インターネット上では IP（Internet Protocol）自体に通信品質を保証する機能がないため，利用者が要求する QoS をエンドツーエンドで保証することは困難である．

このような背景から，インターネット上で QoS を保証する技術として様々な研究開発が行われてきた．IETF では RSVP [1]や DiffServ [2]などの標準化が行わ

† 電気通信大学大学院情報システム学研究科
Graduate School of Information Systems, University of Electro-Communications

本稿の内容は 2000 年 5 月 26 日の DSM 研究会にて報告され，DSM 研究会運営委員により情報処理学会論文誌への掲載が推薦された論文である．

れているし，ネットワーク利用者が動的にネットワークを制御する技術として Active Network[3]の研究がある．しかしこれらの技術が提供する QoS 保証のためのパラメータは，IP 層において QoS 保証を行うためのパラメータなので，利用者がアプリケーションに要求するエンドツーエンドでの QoS を保証するためには，このエンドツーエンドの QoS（以降，アプリケーション QoS と呼ぶ）を IP 層の QoS（以降，ネットワーク QoS と呼ぶ）に変換する必要がある．さらに，一般に各利用者や各アプリケーションはそれぞれに要求する QoS のレベルが異なるため，複数のアプリケーションが共存するシステムでは，これらのアプリケーション間でシステム資源（CPU 時間やネットワーク帯域など）の割当てに対する交渉が必要である．しかし，これらの技術ではこの交渉を実現する手段が提供されていない（以降，この交渉作業のことを QoS 交渉と呼ぶ）．

これらの問題を解決するために，QoS 交渉を実現しながらアプリケーション QoS をネットワーク QoS に割り当てる手法の研究が行われている[4)~7)]．これらの研究では，利用者の QoS に対する要求やシステム環境（ネットワーク上のトラフィック量やネットワークトポロジなど）を入力とし，各アプリケーションへのシステム資源の割当て結果が出力となる最適化問題が解かれている．そのため様々なモデル化が行われ，たとえば，市場モデルを利用したも[6)]，Broker を利用したもの[4)]，マルチエージェントを利用したもの[5),7)]がある．しかし，現在のインターネット環境は新しいアプリケーションが頻繁に追加，削除され，トラフィック需要を予測することは困難になっている．そのため，膨大な数のシステム資源と不確定な需要予測結果を入力に，複雑なモデルを用いて最適解を求める方法を適用するのは困難である．そのためこれらの手法では，実システムへの適用が困難であるという問題と，新たなアプリケーションの追加が困難であるという問題が発生する．

このような観点から本稿では，複雑なモデル化をともなわないフィードバック型の簡易な制御方法を採用する，アプリケーション QoS 管理システム（application QoS Management System: **QMS**）を提案する．**QMS** はアプリケーション QoS をネットワーク QoS へ割り当てることをせずに，システム環境に対して適応的にアプリケーション QoS の制御を行う．ここで，**QMS** は，ある IP ネットワーク（大学の研究室 LAN や企業における部署内 LAN など）を管理する人が，その管理対象のシステムにおいてアプリケーション QoS を管理することを目的に設計されている．管理者の業務としては，システム全体としてのアプリケーション QoS の管理指針（以降，システム運用ポリシーと呼ぶ）を決定すること，システム運用ポリシーを基に各アプリケーションごとに QoS の制御方法を決定するための規則（以降，QoS 管理ポリシーと呼ぶ）を決定すること，これらのポリシーに従って新たなアプリケーションをシステムに導入/削除すること，これらのポリシーの変更を行うことなどを想定している．このような管理者のことを本稿ではアプリケーション QoS 管理者（AP 管理者）と呼ぶ．一般には，システム管理者やネットワーク管理者がこの AP 管理者を兼ねる場合が多い．

このように **QMS** は AP 管理者による利用を前提としているため，システム環境に対して適応的にアプリケーション QoS の制御を行う機能とともに，AP 管理者が QoS 管理ポリシーを **QMS** に設定する作業を支援する機能を持つ．この機能によって，AP 管理者は QoS 管理ポリシーを **QMS** に容易に設定することができ，また複数アプリケーション間での制御の優先順位を容易に決定することができるようになる．その結果，複数アプリケーション間での QoS 交渉が実現し，システム全体として QoS 管理ポリシーに従ったシステムの運用が可能になる．

以下，2 章でアプリケーション QoS 管理システムについて述べ，3 章で **QMS** における QoS の制御方法を説明する．4 章では **QMS** によって複数アプリケーション間での QoS 交渉が実現され，システム全体で QoS 管理ポリシーに従ったシステムの運用が可能になることをシミュレーション実験によって確認する．5 章で関連研究との比較を行った後，6 章で本稿をまとめる．

2. アプリケーション QoS 管理システム

本章では，**QMS** のシステム構成を示し，**QMS** における QoS 制御の流れを説明する．

2.1 システムの概要

QMS は，分散マルチメディアシステムにおいて，アプリケーション QoS を適応的に運用することを目的に設計される．その設計目標は以下の 3 点である．

- アプリケーション QoS を制御の対象とする：

アプリケーション QoS はネットワーク上でのパケット損失やパケット遅延などのネットワーク QoS とは異なり，ビデオの映像品質やシステムの応答時間など，アプリケーションの利用者が体感できる QoS である．つまり，**QMS** ではネットワーク上でパケットの損失が

発生しても，アプリケーションの利用者が満足する品質のサービスが提供されていれば問題はないと考える．

- 実システムへの適用を目指し，複雑なモデル化をともなわないフィードバック型の簡易な制御を行う：**QMS** では QoS の劣化を検出したとき，その QoS に対する要求をネットワーク QoS に割り当てることをせずに，単純なある決まった制御を実施する．たとえば，ビデオの映像品質の劣化を検出した場合，ある決まった帯域ずつビデオの送出帯域を減少させていく制御などを行う．
- 複数アプリケーション間で QoS 交渉を実現する：通常，管理対象のシステム上には複数のアプリケーションが共存しており，それぞれが QoS に対する異なる要求を持っている．**QMS** ではこれらの要求の間で QoS 交渉を実現し，システム全体としてすべてのアプリケーションが QoS 管理ポリシーに従って運用されることを目指す．そのため，管理対象システム上に実装されるすべてのアプリケーションに対して制御の優先順位を決定し，その優先順位に従って制御を行う．

我々はこれらの設計目標を実現するため，**QMS** を以下の 3 種類のモジュールで構成した．アプリケーション QoS を測定，監視する *Notificator*モジュール，QoS 管理ポリシーを保持しこれに従って制御方法を決定する *Manager*モジュール，実際の制御を行う *Controller*モジュールの 3 種類である．**QMS** は管理対象のシステムにおいて，ネットワーク上に分散配置されるこれらのモジュール間の通信によって，アプリケーション QoS の制御を行う．

ここで **QMS** の管理対象のシステムにおける構成要素を定義しておく．管理対象のシステムには，各アプリケーションのサービスを提供する複数のサーバ計算機，各アプリケーションのサービスを受ける複数のクライアント計算機，管理対象のシステムを集中監視するための 1 台の管理用計算機があり，これらの計算機が IP ネットワークによって接続されている．1 台の管理用計算機によって管理が可能な範囲であれば，単一のネットワークセグメントであっても複数のネットワークセグメントであってもかまわない．ただし同一の QoS 管理ポリシーに基づいて管理する必要があるので，管理対象のシステムは同一の管理グループによって管理されている必要がある．次節で **QMS** のシステム構成と各モジュールの機能について説明する．

2.2 システム構成

QMS のシステム構成を**図 1** に示す．**QMS** を構成する 3 種類のモジュールはそれぞれいくつかのオブジェクトから構成され，これらのオブジェクト間の通信によってアプリケーション QoS の制御を行う．このとき実装を容易にするため，オブジェクト間の通信に CORBA（Common Object Request Broker Architecture）[8]などの分散オブジェクト環境を利用する．以下に各モジュールの機能と構成オブジェクトを記す．

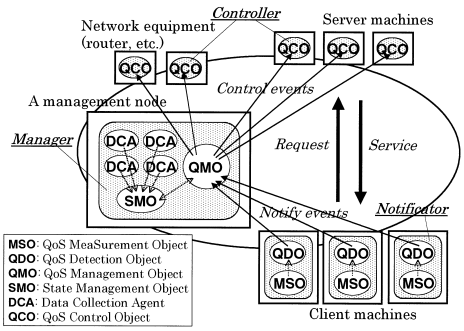

図 1 システム構成
Fig. 1 System architecture.

(1) *Notificator*モジュール

クライアント計算機上（PC，PDA など）に実装され，アプリケーション QoS の劣化を検出する．アプリケーション QoS の測定を行う MSO（QoS MeaSurement Object），QoS 管理ポリシーに基づき QoS 劣化を検出する QDO（QoS Detection Object）の 2 種類のオブジェクトから成る．

(2) *Manager*モジュール

管理対象のシステムに 1 つ存在し，管理用計算機上（WS，PC など）に実装する．管理対象のシステムに実装されるすべてのアプリケーションの QoS 管理ポリシーを保持し制御方法を決定する QMO（QoS Management Object），制御方法を決定するときに必要なデータを収集する DCA（Data Collection Agent），DCA で収集したデータを蓄積する SMO（State Management Object）の 3 種類のオブジェクトから成る．

(3) *Controller*モジュール

制御を実行する対象ごとに存在し，実際の制御を行う．ネットワーク装置（ルータなど）やサーバ計算機上（WS，PC など）に実装する．*Manager*モジュールが決定した制御を実施する QCO（QoS Control Object）1 種類から成る．**QMS** では，各アプリケーションやネットワーク装置が持つ既存の制御機能をできる限り利用する．

2.3 オブジェクト間の通信

QMS におけるオブジェクト間の通信の手順を**図 2** に示す．まず，MSO は QoS パラメータ（映像の品質など）を周期的に測定し，それを QDO に送信する．QDO は，AP 管理者によってあらかじめ設定された

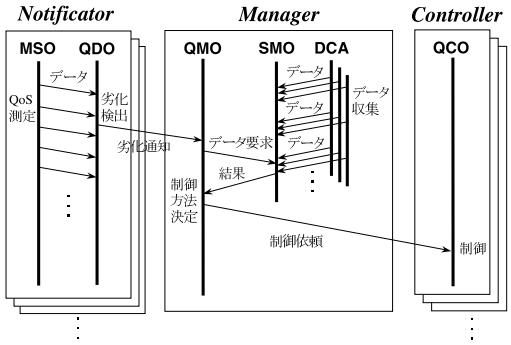

図 2 オブジェクト間の通信
Fig. 2 Procedure in QMS.

しきい値と受信したパラメータを比較し，QoS の劣化を検出する．QDO は，この劣化の検出結果を QMO に通知する．一方，DCA は制御方法を決定するのに必要なデータを各計算機やネットワーク装置から周期的に収集し，その収集したデータを SMO に蓄積する．QDO から劣化の通知を受信した QMO は，AP 管理者が設定した QoS 管理ポリシー，SMO に蓄積された各種データを基に QoS の制御方法を決定し，QCO に実際の制御を依頼する．

QMS の設計目標から，AP 管理者はこの制御方法の決定にフィードバック型の簡易な制御方法を採用する．QMO への QoS 管理ポリシー設定方法については 3 章で詳しく述べる．

3. QMS における QoS 制御

本章では，**QMS** における QoS 制御方法について説明する．さらに，AP 管理者が QoS 管理ポリシーを **QMS** に設定する方法を提案する．

3.1 基本方針

QoS 管理ポリシーの設定は，以下の 2 段階で行う．

- ネットワークレベルの制御の優先順位を決定する．
- アプリケーションレベルの制御方法を決定する．

ここでネットワークレベルの制御とは，IP 層においてネットワーク QoS を保証するために行われる制御のことである．たとえば，ルータのバッファあふれが発生したとき優先順位に従って IP パケットを廃棄する制御などがある．**QMS** における QoS 制御はアプリケーション QoS の制御であるが，複数アプリケーション間で QoS 交渉を実現するために，各アプリケーションのネットワークレベルの制御の優先順位を決定する．このネットワークレベルの制御によって，各アプリケーション QoS の相対的な優先度を決定する．

しかし，相対的な優先度を決定しただけではアプリケーション QoS は保証されない．特にネットワークがふくそう状態になったとき，QoS の保証は困難である．そこで，アプリケーションレベルの制御によってアプリケーション QoS の維持を目指す．ここでアプリケーションレベルの制御とは，サーバの負荷分散を行ったり，映像配信サービスを静止画サービスに変更するなどの各アプリケーションに依存した制御のことである．**QMS** の設計目標から，このアプリケーションレベルの制御では，フィードバック型の簡易な制御方法を採用する．

このように **QMS** では，ネットワークレベルの制御によって QoS 交渉を実現しつつ，アプリケーションレベルの制御によって各アプリケーション QoS の維持を目指す．その結果，アプリケーション QoS をネットワーク QoS に割り当てることなしに，アプリケーション QoS の制御が可能になる．以下，ネットワークレベルの制御とアプリケーションレベルの制御の詳細，および各制御において QoS 管理ポリシーを **QMS** に設定する方法について述べる．

3.2 ネットワークレベルの制御

(1) 制御方法

QMS のネットワークレベルの制御では，インターネットにおけるエンドツーエンドサービス品質として規定される，スループット，IP データグラムの遅延，遅延ゆらぎ，損失の 4 種類のデータ[9]を制御対象とする．これら 4 種類の QoS パラメータを制御する方法として，それぞれ以下の制御を行うこととした．

- スループットの確保：RSVP などを利用した転送帯域の保証を行う．これはスループットが劣化する要因として，通信相手先に到達するまでの通信経路の違い，経路上のルータの性能とその数，ルータ間の回線速度とその混雑度合いなどが考えられるためである．
- IP データグラムの遅延：CBQ（Class Based Queuing）などを利用したルータのキュー管理を行う．これは遅延の発生する要因として，ルータにおけるルーティングテーブルの検索時間，回線伝播遅延などが考えられるためである．
- IP データグラムの遅延ゆらぎ：トークンバケット方式などを利用したトラフィックシェーピングを行う．これは遅延ゆらぎの発生する要因として，ルータにおけるルーティングテーブルの検索時間，回線伝播遅延などが考えられるためである．
- IP データグラムの損失：DiffServ などを利用したパケット廃棄の優先度の設定を行う．これは損失の発生する要因として，ルータのバッファあふれなどが考えられるためである．

QMS では，管理対象のシステムに実装されるすべ

てのアプリケーションに対して，これら 4 種類の制御における優先順位をあらかじめ決定しておく．この優先順位は，各アプリケーションのトラフィック特性によって **QMS** がデフォルトの順位を決定し，AP 管理者がシステム運用ポリシーに応じてその順位を変更することによって決定される．この 4 種類の制御に対する優先順位が，ネットワークレベルにおける QoS 管理ポリシーになる．

実際のネットワークレベルの制御はルータなどのネットワーク装置で実施される．そのため **QMS** は，ネットワークレベルの QoS 管理ポリシーが **QMS** に設定されると，その優先順位に従った制御方法を各ネットワーク装置に設定する．

(2) QoS 管理ポリシーの設定方法

前項で述べたように，ネットワークレベルの制御の優先順位は，各アプリケーションのトラフィック特性に応じて **QMS** がデフォルトの順位を決定する．**QMS** はこのデフォルトの優先順位を AP 管理者に提示し，AP 管理者は必要に応じてこの順位を変更する．ここで各アプリケーションのトラフィック特性は，管理対象のシステムに新しいアプリケーションを追加するときに AP 管理者が **QMS** に設定する．このアプリケーションのトラフィック特性に応じた分類方法，およびこの特性に応じてデフォルトの優先順位を決定するための規則について以下に述べる．

アプリケーションの分類 マルチメディアサービスにおけるトラフィック特性は，トラフィックの種類と応答時間に対する要求によって分類される[10]．アプリケーションから発生するトラフィックの種類を**表 1** に，応答時間に対する要求種別を**表 2** に示す．**QMS** では，各アプリケーションをこれらのトラフィック特性に従って分類し，分類ごとにデフォルトの制御方法とその優先順位を決定する．

デフォルトの QoS 管理ポリシーを決定するための規則 **QMS** では，表 1，表 2 の特性によって分類されたアプリケーションに対し，**表 3** に示す規則に従って制御の優先順位を決定する．つまり QMO の内部にこの表を保持し，AP 管理者が各アプリケーションのトラフィック特性を指定することによって，各制御の優先順位を QoS 管理ポリシーとして決定する．ここで，表 1，表 2 の分類を組み合わせると 9 種類の分類があるが，S-NRT 型と S-QRT 型に該当するアプリケーションは存在しないので，表 3 に示す 7 種類の分類に対して優先順位を決定した．

QMS は，この規則に従って **QMS** が推奨するデフォルトの優先順位を決定し，その結果を AP 管理者に提示する．AP 管理者はこの結果を参考に，システム運用ポリシーに応じて順位の変更を行う．QMO はその変更した結果をネットワークレベルの QoS 管理ポリシーとして保持し，同時にこの QoS 管理ポリシーをネットワーク装置に実装可能な制御方法に変換して，ネットワーク装置に設定する．

表 1 トラフィックの種類による分類
Table 1 Application classification (traffic types).

トラフィックの種類	記号	特徴	AP 例
ブロック型	B	大きいサイズの情報が低い頻度で発生	静止画
ストリーム型	S	中サイズの情報が周期的に発生	映像
トランザクション型	T	小サイズの情報が不規則に発生	コマンド

表 2 応答時間に対する要求による分類
Table 2 Application classification (response time).

応答時間	記号	特徴	AP 例
リアルタイム	RT	数 100 ms 以下が必要	電話
準リアルタイム	QRT	数秒以下が望ましい	WWW
非リアルタイム	NRT	100 秒以下も許容	e-mail

表 3 QoS 管理ポリシーを決定する規則
Table 3 The rule setting QoS management policies.

分類	帯域制御	キュー送出順位	シェーピング	パケット廃棄
B-RT	1	0	1	1
S-RT	0	0	0	0
T-RT	なし	0	なし	4
B-QRT	なし	1	なし	2
T-QRT	なし	1	なし	4
B-NRT	なし	2	なし	3
T-NRT	なし	2	なし	4

(3) ネットワーク装置への制御方法の設定

QMO に保持されたネットワークレベルの QoS 管理ポリシーは，QMO でネットワーク装置に実装可能な制御方法に変換され，QMO がネットワーク装置に設定する．このときネットワーク装置への設定に必要なパラメータは，あらかじめ QMO に設定されている必要がある．そのため，管理対象のシステム内に新規アプリケーションを追加する場合，これらのパラメータを合わせて QMO に設定する．設定が必要なパラメータを以下に示す．

- アプリケーション名
- サーバ計算機とクライアント計算機の IP アドレス
- サーバからの送出帯域（平均値，最大値）
- ポート番号とポート種別（TCP/UDP/Other）
- ネットワークトポロジ
- ネットワーク装置の IP アドレス

QMO はこれらのパラメータを利用して，ネットワー

ク装置に制御方法を設定する．各制御に対する設定方法を以下に述べる．

帯域保証 サーバ計算機の平均送出帯域を予約帯域と見なし，クライアント計算機の IP アドレスを利用して，クライアント計算機に予約帯域を設定する．複数のアプリケーションが帯域保証を要求している場合，優先順位に従って順番に予約を行う．

キュー管理 ネットワーク装置の IP アドレス，各アプリケーションのポート種別とポート番号を取得し，ポートによって識別されるアプリケーションとそのアプリケーションがどの分類に入るかという対応表，および分類されたアプリケーションの集合とその振舞（送出優先順位）の対応表をネットワーク装置に設定する．分類の数はネットワーク装置に依存するが，制限がない場合は **QMS** における優先順位の数とする．

シェーピング サーバ計算機の最大送出帯域，ポート種別とポート番号，出口ノード（ネットワーク装置）の IP アドレスを取得し，帯域とポートの対応表を出口ノードに設定する．複数アプリケーションがシェーピングを要求している場合，優先順位に従って順番に設定を行う．

パケット廃棄 ポート種別とポート番号，入口ノード（ネットワーク装置）の IP アドレスを取得し，ポートによって識別されるアプリケーションとそのアプリケーションがどの分類に入るかという対応表，および分類されたアプリケーションの集合とその振舞（廃棄優先順位位）の対応表を入口ノードに設定する．

(4) 実装例

ネットワークレベルの QoS 管理ポリシーを **QMS** に設定する機能を，Java アプリケーションにより実装した．実装例を**図 3**，**図 4** に示す．新しいアプリケーションを追加する場合，AP 管理者は新規アプリケーション追加画面においてアプリケーション名を入力し，トラフィックの種類と要求される応答時間を選択メニューから選ぶ．すると **QMS** は，前節で規定した優先順位を決定する規則に従い，ネットワークレベルの制御の優先順位を決定する．AP 管理者は，この優先順位を 4 種類の変更画面で確認し，必要に応じて順位の変更を行う．ここで決定した優先順位は，QMO がネットワーク装置における制御方法に変換し，各ネットワーク装置に設定する．

本機能により，AP 管理者はアプリケーションのトラフィック特性を選択するだけで，容易にデフォルトの QoS 管理ポリシーを設定できるようになる．また，4 種類の制御の優先順位を決定することにより，既存のシステムに新規アプリケーションを追加した場合でも，容易に QoS 交渉が実現できる．今回は実装していないが，これらの機能のほかにアプリケーションごとの TCP/UDP ポートなど，制御に必要な各種パラメータを設定する機能が必要である．

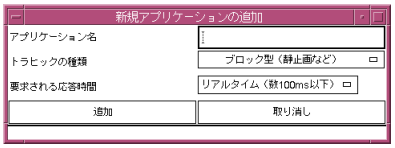

図 3 新規アプリケーション追加画面
Fig. 3 The window in adding new application.

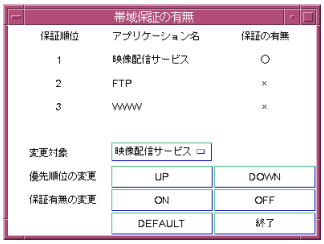

図 4 優先順位変更画面
Fig. 4 The window in changing the priority.

3.3 アプリケーションレベルの制御

(1) 制御方法

アプリケーションレベルの制御対象は，システム運用中に準リアルタイムで変更可能な要素とする．これは **QMS** における QoS 制御が，フィードバック型の簡易な制御であることによる．**QMS** では以下の 4 種類の制御を行うが，制御方法を決定するときの基本方針は，フィードバック型の簡易な制御方法を採用すること，および QoS 劣化の要因のないときには通常のサービスを提供するが，品質が劣化した場合に犠牲にできるものを決定することである．

- サービスレベルの変更：アプリケーションが提供するサービスの品質を変更する制御．たとえば，映像配信サービスにおいて映像送出帯域を変更する制御，IP 電話においてエンコードレートを下げる制御などがある．フィードバック型の制御方法を採用するため，劣化を検出したらある単位ずつサービス品質を低下させ，劣化の回復を検出したらある単位ずつサービス品質を回復させていく制御を採用する．
- サーバの負荷分散：サーバ計算機の負荷分散を行う制御．簡易な制御方法を採用するため，ある時点で最も負荷が少ないサーバを選択するという制御や，ラウンドロビンによって順番に接続先サーバを変更していく制御などを採用する．
- エラー訂正：アプリケーションにもともと含まれているエラー訂正機能を利用する制御．冗長パケット

によるエラー検出などを行う．

● 提供サービスの変更：アプリケーションが提供するサービスの内容を変更する制御．たとえば，テレビ会議サービスにおいて映像を静止画に変更する制御，WWW サービスにおいて画像をテキストに変更する制御などがある．劣化を検出したらサービス内容を変更し，劣化の回復を検出したらサービス内容を元に戻す制御を採用する．

QMS は QoS の劣化を検出すると，これら 4 種類の制御のうちの 1 つ，または複数を組み合わせて実施する．AP 管理者は 4 種類の制御のうち実装可能な制御をすべて実装しておくが，実際に実行する制御は運用ポリシーに従って 4 つの中からいくつかを選択しておく．

(2) QoS 管理ポリシーの設定方法

アプリケーションレベルの QoS 管理ポリシーは，AP 管理者がアプリケーションごとに独立に設定する．以下にその手順を示す．

- AP 管理者はアプリケーション QoS を決定し，MSO，QDO としてオブジェクト化する．
- AP 管理者は，4 種類の制御のうち実装可能な制御を QCO としてオブジェクト化する．
- AP 管理者は，アプリケーションレベルの制御方法を決定し QMO に実装する．制御におけるフィードバックパラメータは，シミュレーション実験などによって事前に決定しておく．
- AP 管理者は，MSO，QDO，QCO を *Manager* モジュール内の指定ディレクトリに置く．
- **QMS** は，QMO に設定されたサーバ計算機とクライアント計算機の IP アドレスに従って，MSO，QDO，QCO を各計算機に実装する．このとき AP 管理者は，4 種類の制御のうち実装する制御を選択する．

4. シミュレーション実験

QMS によってシステム全体で QoS 管理ポリシーに従ったシステム運用が可能になることを確認するため，シミュレーションによる検証実験を行った．本章では，その実験結果について述べる．

4.1 実験条件

本実験では，1 台の計算機上に擬似的にネットワークを構築し，擬似トラフィックを発生させることによってネットワーク上での各アプリケーションの動作を模擬した．実験で利用したネットワーク構成を**図 5** に示す．また，実験の対象となるアプリケーションの種類とトラフィックの形態，実装した計算機などを**表 4** に示す．ここで各アプリケーションの送出帯域は一定値，ストリーム型サービスである映像配信のサービス時間は平均 600 秒のランダム値，各アプリケーションの送信間隔は表に示す平均を持つランダム値とした．

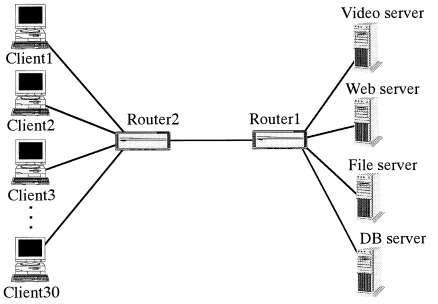

図 5 シミュレーション実験におけるネットワーク構成
Fig. 5 Network architecture for the simulation.

表 4 アプリケーションの種類
Table 4 Target applications.

AP 名	分類	帯域 (一定)	送信間隔 (平均)	実装 計算機
映像配信	S-RT	10M	100 s.	1〜20
WWW	B-QRT	1M	100 s.	21〜30
FTP	B-QRT	30M	200 s.	21〜30
DB アクセス	T-QRT	10k	50 s.	21〜30

4.2 QoS 管理ポリシーの設定

提案した QoS 管理ポリシーの設定機能を利用し，上記の 4 種類のアプリケーションに対する QoS 管理ポリシーを決定する．今回の実験では 3 種類のシステム運用ポリシーを設定し，**QMS** によって様々なポリシーを満足するシステム運用が可能になることを示す．

運用ポリシー 1 トランザクション型アプリケーションを優先する．本実験では，DB アクセスの可用性，応答時間を最優先とする．

運用ポリシー 2 特定の利用者のサービスを優先する．本実験では，FTP クライアントのうち 2 台のデータの信頼性，応答時間を最優先とする．

運用ポリシー 3 ストリーム型サービスを優先し，ネットワークに高負荷をかける．本実験では，映像クライアントのうち 7 台の遅延防止，映像品質を最優先とする．

次に，各システム運用ポリシーにおけるネットワークレベルの制御の優先順位，およびアプリケーションレベルの制御方法を決定する．今回のシミュレーション実験では，帯域保証およびシェーピングは行わなかった．各ポリシーごとのキュー管理の優先順位，およびパケット廃棄の優先順位を**表 5**，**表 6** に示す．アプリケーションレベルの制御方法は全運用ポリシー共

通で，映像の送出帯域の変更を行った．これは，映像配信サービスにおいて映像品質の劣化を検出するたびに，送出帯域を 1 Mbps ずつ下げていくことによって実現した．劣化が回復した場合，次周期での送出帯域を，10 Mbps に達するまで 1 Mbps ずつ上げていく．実環境の場合，映像品質の劣化は利用者（またはクライアント計算機）によって検出が可能であるが，今回のシミュレーション実験では，パケット損失の検出を映像品質の劣化と見なし制御を行った．

4.3 実験結果

シミュレーション時間で 1,800 秒の間に，各アプリケーションが要求した帯域，および実際にクライアント計算機が受信した帯域を測定した．実験では比較のために，制御をまったく行わなかった場合（制御なし），およびネットワークレベルの制御のみを行った場合（NW のみ）の値も測定した．実験結果を**表 7** に示す．表中の値はデータ受信率（受信帯域/要求帯域）を，カッコ内の値はシミュレーション時間中の総要求帯域（単位は Gbps）を示す．また各運用ポリシーにおいて優先されるアプリケーションのデータには，* をつけた．

制御を行わない場合，どの運用ポリシーのどのアプリケーションにおいても高い割合でデータ損失が発生している．この場合，ネットワーク装置ではアプリケーションの種類を特定できないので，ネットワークの品質劣化が始まるとどのアプリケーション QoS も同じ割合で劣化する．特にデータ損失によってサービスの提供自体が不可能になる DB アクセスや FTP では，アプリケーション QoS の劣化が顕著になる．つまり，制御をまったく行わないと，多くの帯域を必要とするアプリケーション（この場合，映像配信サービス）が他のアプリケーションを圧迫し，システム全体としてアプリケーション QoS が劣化することが分かる．

ネットワークレベルの制御を行った場合は，映像配信（非優先）以外のアプリケーションではほとんどデータ損失は発生していない．つまりネットワークレベルの制御によって，各運用ポリシーにおいて優先度の高いアプリケーションの品質が維持されていることが分かる．しかし，優先度の低い映像配信サービスでは多くのデータ損失が発生している．特に広帯域を必要とする映像配信サービスを優先する運用ポリシー 3 においては，非優先の映像配信サービスの QoS 劣化が著しい．このように，ネットワークレベルの制御によって優先度の高いアプリケーションの品質は維持できるが，広帯域を必要とするアプリケーションを優先するなどふくそう状態が引き起こされる運用ポリシーが設定された場合，システム全体で QoS 管理ポリシーを満たした制御を行うことができない．特に，優先度の

表 5 キュー管理の優先順位
Table 5 The list of priority for packet sending.

AP 名	ポリシー 1	ポリシー 2	ポリシー 3
DB アクセス	0	1	1
FTP (非優先)	1	3	2
FTP (優先)	–	0	–
WWW	2	2	3
映像配信 (非優先)	3	4	4
映像配信 (優先)	–	–	0

表 6 パケット廃棄の優先順位
Table 6 The list of priority for IP packet discard.

AP 名	ポリシー 1	ポリシー 2	ポリシー 3
映像配信 (非優先)	0	0	0
映像配信 (優先)	–	–	4
WWW	1	2	1
FTP (非優先)	2	1	2
FTP (優先)	–	4	–
DB アクセス	3	3	3

表 7 シミュレーション実験で測定されたデータ受信率
Table 7 Data receiving ratio on the simulation experiment.

サービス名	ポリシー 1			ポリシー 2			ポリシー 3		
	QMS	制御なし	NW のみ	**QMS**	制御なし	NW のみ	**QMS**	制御なし	NW のみ
映像配信 (非優先)	0.89	0.53	0.52	0.89	0.52	0.52	0.73	0.52	0.26
	(191)	(331)	(328)	(192)	(335)	(331)	(79)	(215)	(210)
映像配信 (優先)	–	–	–	–	–	–	1.0*	0.54*	1.0*
	–	–	–	–	–	–	(114)	(116)	(116)
WWW	1.0	0.53	1.0	1.0	0.50	1.0	0.92	0.51	0.91
	(0.35)	(0.36)	(0.36)	(0.35)	(0.36)	(0.36)	(0.35)	(0.37)	(0.36)
FTP(非優先)	1.0	0.53	1.0	1.0	0.41	1.0	0.95	0.52	0.94
	(5.4)	(5.3)	(4.9)	(4.1)	(4.4)	(4.6)	(5.1)	(5.1)	(5.1)
FTP(優先)	–	–	–	1.0*	0.51*	1.0*	–	–	–
	–	–	–	(1.1)	(1.2)	(1.1)	–	–	–
DB アクセス	1.0*	0.53*	1.0*	1.0	0.50	1.0	1.0	0.46	1.0
	(0.007)	(0.007)	(0.007)	(0.007)	(0.007)	(0.007)	(0.007)	(0.007)	(0.007)
平均受信率	0.97	0.53	0.86	0.98	0.49	0.90	0.92	0.51	0.82

低いアプリケーションの QoS が著しく劣化する．

QMS では，アプリケーションレベルで映像配信サービスの制御を行った結果，送出帯域を減少させることによって映像配信サービスでのデータ損失が大きく減少している．データ損失による映像品質の劣化より送出帯域の減少による劣化の方が少ないと考えられるため，この制御によって優先度の低いアプリケーションの QoS 向上が期待できる．これは 3 種類の運用ポリシーすべてに対していえることで，つまり，システム全体で QoS 管理ポリシーに従った運用が実現することが分かった．ここでの制御はフィードバック型の簡易な制御だが，ふくそう状態時にも QoS 交渉がうまく働いていることが分かる．

5. 関連研究との比較

本章では，**QMS** と関連研究との比較を行う．まず IP ネットワーク上でアプリケーション QoS を制御する研究としては，RSVP [1] や DiffServ [2] のようにネットワーク QoS を制御することによってエンドツーエンドの品質を維持しようとする研究がある．これらの研究ではその制御対象がネットワーク QoS であるため，システム管理者の勘と経験によってアプリケーション QoS をネットワーク QoS に割り当てる必要がある．**QMS** でもこれらの技術をネットワークレベルの制御手法として利用するが，アプリケーションごとのトラフィック特性から制御の優先順位を決定すること，およびアプリケーションレベルの制御を組み合わせて行うところがこれらの研究と異なる．

次にアプリケーション QoS をネットワーク QoS に割り当てる研究として，市場モデルを利用したもの[6]，ブローカを利用したもの[4]，エージェントを利用したもの[5),7]などがあるが，これらの研究が割当てのために複雑なモデル化を必要とするのに対し，**QMS** はネットワークレベルの制御を組み合わせることによってフィードバック型の簡易な制御方法を採用することができる．簡易な制御であってもシステム全体として QoS 管理ポリシーに従った運用が可能になることは，4 章で確認した．

ここではその割当てにおける計算量，および新規アプリケーションを追加する方法に関する考察を行う．まず割当てにおける計算量では，市場モデルを利用した研究[6]の場合，市場指向プログラミング環境 WALRAS を利用しているので，計算量は WALRAS の計算量に依存する．文献中では，この計算時間がシステム動作上無視できない程度に大きいことが示されている．またエージェントを利用した研究[7]の場合，QoS 交渉において任意の妥協基準に基づいて妥協品質の計算を行う．文献中では，優先度の低いアプリケーション間で交渉を行わせるには（つまり交渉コストを低くするには），QoS 調整時間が非常に長くなることが示されている．一方，**QMS** ではフィードバック型の簡易な制御方法を採用するため，計算量はパラメータに対する線形時間でおさえられる．

次に，管理対象のシステムに新規アプリケーションを追加する方法について考える．市場モデルを利用した研究[6]では，構築された市場モデルに新しい Consumer と Producer を追加し，追加された要素に関する新たな財の流れを追加する必要がある．これはモデルの作り直しを意味し，新しいアプリケーションが頻繁に追加，削除される系への適用は難しい．またエージェントを利用した研究[7]では，新しいアプリケーションを追加するたびに新たな QoS 割当てモデルを作成し，システムに追加する必要がある．そのため，新規アプリケーションの追加は困難である．一方，**QMS** における新規アプリケーションの追加方法は，アプリケーションに対応する新規オブジェクトの追加，およびオブジェクトの属性値追加によって実現される．また，3 章で記述した新規アプリケーション追加機能によって，容易に新規アプリケーションが追加できる仕組みがシステムに組み込まれている．

ただし **QMS** のアプリケーションレベルの制御では，たとえば劣化の検出ごとに 1 単位ずつ帯域を下げるようなフィードバック型の制御を行うと，制御の結果が振動し収束しない可能性がある．これに対しては，ネットワークレベルの制御を組み合わせることによって対処する．また，つねにふくそう状態にある系に対して優先度の低いアプリケーションがサービス不能に陥る危険はあるが，そのような慢性的な資源不足には設備増加で対処すべきと考える．

6. ま と め

本稿では，分散マルチメディアシステムの適応的運用に着目し，フィードバック型の簡易な制御を実現するアプリケーション QoS 管理システム（**QMS**）の提案を行った．**QMS** における QoS 制御は，アプリケーション QoS を基に決定されるアプリケーションレベルの制御方法と，アプリケーションのトラフィック特性を基に決定されるネットワークレベルの制御方法を組み合わせて実施される．その結果，複数アプリケーション間の QoS 交渉が実現し，システム全体として QoS 管理ポリシーに従ったシステム運用が可能になる．本稿ではさらに，3 種類のシステム運用ポリシー

に対し，**QMS** によって QoS 管理ポリシーに従ったシステム運用が可能になることをシミュレーション実験結果として示した．特にふくそう時に相対的に優先度の低いアプリケーションの QoS が向上することを確認した．

最後に，**QMS** の問題点と今後の課題をここにまとめておく．**QMS** の設計目標の 1 つである実システムへの適用を考えた場合，解決しなくてはならない問題は 2 つある．まず，既存のアプリケーションへの **QMS** の適用方法についてである．本稿ではシミュレーション実験によって仮想的なシステム上に **QMS** を適用したが，既存のアプリケーションに **QMS** を適用する場合，QoS 劣化の検出方法，各種性能データの収集方法，制御の実現方法，分散オブジェクト環境を利用することによる実装性能への影響など，様々な研究課題が残っている．我々は現在，大学の研究室内にマルチメディアサービスを実装した実験システムの構築を進めており，今後そこに **QMS** を実装することによって，**QMS** の適用実験，実装性能評価などを行っていく予定である．**QMS** は複数のオブジェクトから構成され，今回利用したシミュレータもこのオブジェクト単位で実装しているので，これをできる限り再利用して実験システムを構築する予定である．

もう 1 つの問題は規模適応性に関するものである．本稿では同一の管理グループによって管理されるネットワークシステムを制御対象としたが，**QMS** をインターネット上での応用を考えたシステムに適用する場合，異なる管理グループによって管理されるネットワークシステムの間での QoS 交渉の方法や，複数の *Manager*モジュール間の連携方法などの検討が必要になってくる．**QMS** はアプリケーションレベルの制御方法に関しては，フィードバック型の簡易な制御方法を用いるため規模適応性に対応できると考えられる．しかし，ネットワークレベルの制御方法の中には RSVP のように規模適応性に欠ける方法も存在するので，どのような範囲でどのような制御をどのように組み合わせて適用すべきかを検討する必要がある．さらに，セグメントをまたいだ通信では，同じアプリケーションでもセグメントによって QoS 管理ポリシーが異なるのがふつうである．その場合，他のセグメントにおける QoS をどのように保証するかという問題が出てくる．インターネットの自律分散性を考慮すると，自セグメント内のみで保証を行い，他セグメントにおいては必要最小限の保証を行うモデルを構築するべきであろう．現在でも，自セグメント内では RSVP などで帯域保証を行い，他セグメントを経由するトラフィックに関しては DiffServ などで対処する手法が提案されている[11]．ここにさらに，**QMS** における「優先度の低いアプリケーションの品質を故意に劣化させる」という考え方を導入する必要であると考える．そのため今後は，複数のセグメントから成るネットワーク構成に対してシミュレーション実験を行っていく予定である．

参 考 文 献

1) Braden, R., et al.: *Resource reSerVation Protocol (RSVP)*, IETF RFC2205 (1997).
2) Brake, S., et al.: *Architecture for Differentiated Services*, IETF RFC2475 (1998).
3) Tennenhouse, D.L., Smith, J.M., Sincoskie, W.D., Wetherall, D.J. and Minden, G.J.: A survey of active network research, *IEEE Commun. Mag.*, Vol.35, No.1, pp.80–86 (1997).
4) Nahrstedt, K. and Smith, J.M.: The QoS broker, *IEEE MultiMedia*, Vol.2, No.1, pp.53–67 (1995).
5) 橋本浩二，野村尚央，柴田義孝，白鳥則朗：QoS 保証を考慮したやわらかいマルチメディアシステム，情報処理学会論文誌，Vol.40, No.1, pp.113–123 (1999).
6) 八槇博史，ウェルマン, M.P., 石田　亨：市場モデルに基づくアプリケーション QoS の制御，電子情報通信学会論文誌 D-I，Vol.J81, No.5, pp.540–547 (1998).
7) Yamazaki, T. and Matsuda, J.: Adaptive QoS management for multimedia applications in heterogeneous environments, *IEICE Trans. Commun.*, Vol.E82-B, No.11, pp.1801–1807 (1999).
8) Object Management Group: *The Common Object Request Broker: Architecture and Specification, Object Management Group*, Framingham, Massachusetts (1998).
9) Paxson, V.: Towards a framework for defining Internet performance metrics, *Proc. INET'96*, p.3 (1996).
10) 間瀬憲一（著），電子情報通信学会（編）：マルチメディアネットワークとコミュニケーション品質，コロナ社 (1998).
11) Stoica, I. and Zhang, H.: Providing Guaranteed Service Without Per Flow Management, *Proc. ACM SIGCOMM'99* (1999).

(平成 13 年 1 月 26 日受付)
(平成 13 年 9 月 12 日採録)

推 薦 文

インターネット環境において，今後はさらに基幹業

務系のアプリケーションサービスとカジュアルな利用との共存が増大すると考えられている．本発表はIP ネットワーク上でサービスを提供する様々なアプリケーションのサービス品質を確保するためのアプリケーション QoS の管理手法を提案している．アプリケーション QoS の管理ポリシーの設定，その制御方法，および簡単なシミュレーション評価により，その実現性，有効性を評価しており，実用化が期待される．

（DSM 研究会運営委員　松浦　敏雄）

加藤　由花（学生会員）

1989 年東京大学理学部卒業．同年日本電信電話（株）入社．ATM 網のトラフィック制御に関する研究，双方向マルチメディアシステムの研究開発に従事．1999 年電気通信大学大学院情報システム学研究科博士前期課程修了．現在同後期課程在学中．分散システムにおける QoS 制御技術の研究に従事．電子情報通信学会，IEEE 各会員．

箱崎　勝也（正会員）

1941 年生．1963 年九州大学工学部電子工学科卒業．同年日本電気（株）入社．中央研究所，ソフトウエア開発グループにおいて，システム性能評価，コンピュータアーキテクチャ，OS，ネットワークの相互接続性等の研究開発に従事．1994 年から電気通信大学大学院情報システム学研究科教授．工学博士．分散システム技術，マルチメディア利用技術，モバイルコミュニケーション，情報システムアーキテクチャ等の研究に従事．情報処理学会論文賞受賞（1982）．分散システム運用技術研究会主査．電子情報通信学会，IEEE，ACM 各会員．