分散システム運用技術　10－6
（１９９８．　７．　２４）

インターネットにおける協調管理プラットフォームの提案

浜田雅樹　　藤崎智宏　　犬束 敏信　　蔭山 克禎
NTT ソフトウェア研究所

概要

インターネットにおける協調管理を支援するプラットフォームの必要性を述べる．インターネットに特徴的な管理体制として，管理領域がオーバーラップする「協調管理」がある．既存の支援システムが前提としている集中型や階層型の管理モデルと異なるため，管理対象の負荷を増大させたり，作業の分担，引き継ぎがしにくいという問題が発生している．これらを解決するため，協調管理プラットフォームは，個別に導入された管理アプリケーションの管理トラヒックを自動的に集約し，また管理アプリケーションが保持する管理の状態に関する情報を，他の管理アプリケーションと共有する仕組みを提供する．

Developing A Platform for Cooperative Internet Management

Masaki *HAMADA*　Tomohiro *FUJISAKI*　Toshinobu *INUZUKA*　Katsuyoshi *KAGEYAMA*
NTT Software Laboratories

Abstract

The necessity of a platform that supports cooperative Internet management is discussed. Internet management is characterized by "cooperative management" where the network is managed by two or more operation centers. Its mismatch to the central or hierarchical management model adopted by existing network management systems causes many problems such as load increase of managed devices and difficulty of sharing management status of the network. To solve these problems, the platform is expected to summarize network management traffics and to facilitate sharing management status information hold by the management application.

1　はじめに

通信ネットワークは，重複がない管理領域に分割して管理するのが効率的と考えられている．領域毎に独立した組織がそれぞれ管理システム(Network Management System, 以下 NMS と記す)を利用して管理する場合もある．その上に広域的管理を行う組織／システムを積み上げていけば，規模の大きいネットワークを管理することが可能になる(階層型の管理形態)．一社が提供することが多い電話網はこの体制をとり易い．しかし，インターネットやイントラネットでは管理組織間で管理領域のオーバーラップがある複雑な形態をとる場合がある．例えば，ネットワーク管理をアウトソーシングをしているが，非常時には依頼元が対応するような場合である．NMS は，アウトソーシング先／元ともに設置し同じ領域を管理している．以下，このような管理領域がオーバーラップする管理形態を「協調管理」と呼ぶ．

既存の NMS は，単純な管理モデル(集中 or 階層型)に基づいており，上記管理形態に利用すると，管理対象の負加

を増加させたり，互いの管理の状態を把握できない問題が発生する．

本論文では，このような問題点を解決する協調管理プラットフォームを提案する．

2 協調管理支援の必要性

2.1 協調管理の特徴

協調管理を，二つ以上の管理組織がネットワークの一部を交互または協力して管理することと定義する．ここで管理とは，監視と制御(故障対応)を指す．

例えば，ネットワーク管理をアウトソーシングしている例を考える．アウトソーシングを受けた組織は，異常がないか定常的に監視し，簡単な故障に対応する．しかし，複雑な問題の場合は，依頼元の組織で対応する必要がある．そのため，NMS は両組織に設置する．

その他にも以下のような例が存在する．

- ネットワークサービスプロバイダ等において，障害の発見や受付を主目的とする常時監視の組織と，複雑な問題が発生した場合に対応するエキスパートの組織が連携する．それぞれが NMS を利用している．
- 終夜監視が必要なネットワークを(時差を利用して)日米間で交替して管理する．
- 各組織は自ネットワークを管理し，かつ相互連結部分(バックボーン等)を別の独立した組織が管理する．

協調管理で利用する複数の NMS の機能，設定は必ずしも一致しない．これは，それぞれの役割，設備，スキルやポリシーが異なるためである．また，NMS 以外に，ルーティングチェッカーやトラヒック測定システム等(NMS と合わせて管理アプリケーションと総称する)を利用する場合もある．このように，ある程度の自由度を有する組織が協調して管理することになる．その中で最低限以下の項目についてコントロールが必要である．

(1) 排他制御

ネットワーク機器の制御に対する権限等の方針を決め，場合によっては排他制御が必要となる．

(2) 管理状態の相互参照や引き継ぎ

例えば，ネットワーク機器に起こった変化に対して(ネットワーク機器から NMS へトラップが送られ通知される，詳細は 2.2 節を参照)，管理者は分析・対処しなければならない．その結果・進捗(例えばトラップを消去する等)を協調管理する組織間で共有する必要がある．

(3) 管理情報の統一

MIB ・トラップの意味付けや定義がそれぞれの管理組織(管理アプリケーション)間で統一されている必要がある．

これらは集中や階層的な管理形態に比べた場合の協調管理の特徴となっている．次に，既存の NMS を協調管理で利用する場合の問題点について議論する．

2.2 既存の NMS との適合性

現在インターネットで用いられている NMS は，集中型か階層型(統合型)の管理形態に基づいている．これらは一般に:

- 種々のネットワーク機器から情報を収集し，
- それらを解析，提示し，
- オペレータがネットワーク機器をコントロールする作業

を支援する．

インターネットの管理プロトコルとして，SNMP (Simple Network Management Protocol)(1)が多く使われている．SNMP は，シンプルなトラップ&ポーリングという管理モデルを持つ．ポーリングとは，コンピュータ NMS が，必要な情報をネットワーク機器から取得する動作である．逆に，トラップは，ネットワーク機器が，予め決められた条件(例えばリンクが切れた)が満たされたとき，その旨を通知するため NMS に向かって送られるものである．

集中型の NMS は，このモデルを実現したものである(図 1)．複数の集中型 NMS を束ねる統合管理システムを設け，階層型の管理体制を構築する場合もある．

協調管理のため，これらを同じ管理領域を管理するように設定して用いた場合，2.1 節で述べた(1) 排他制御，(2)管理状態の相互参照や引き継ぎ，(3)管理情報の統一がシステムでサポートされないため，煩雑な手作業が必要になる．更に各 NMS が SNMP パケットを出すため，管理対象にかかる負荷が増える問題も存在する[1]．

2.3 支援システムへの要求

協調管理では，排他制御と管理情報の統一が行われている管理システム間で，管理状態の相互参照や引き継ぎが

---

[1]NMS の製品によっては，分散管理機能を用いて，SNMP パケットを重複して出さない構成をとれる．但し，機能は限定される(ネットワークのトポロジ監視のみ等)(6)．

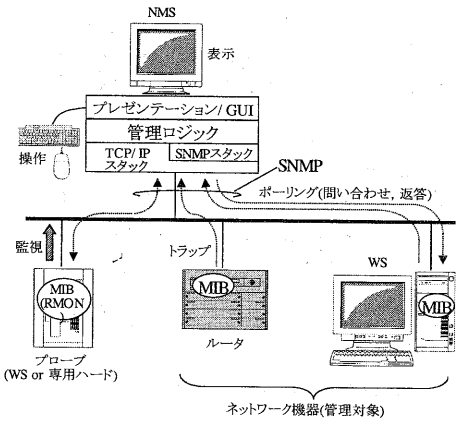


図 1　SNMP による集中型管理システム

できる機構が必要になる. また, できるだけ管理対象に負荷をかけない工夫が必要である.

このような機構を持つ管理システムを各自が開発するのはコストと時間がかかり好ましくない. インターネットは厳しい競争の中デファクトスタンダードを中心に進む傾向にある. 管理アプリケーションも例外ではなく, 魅力的な良い製品がある. しかし, 任意に選んだ製品を連携させるには, それぞれ個別の API を用いたプログラムを開発する必要がある. 現在, これらの横断的に利用できる標準は SNMP しかない. 市場にある優れた製品を使い協調管理するには, これらの製品を連携させる糊付けの役割を果たす「協調管理プラットフォーム」を実現することが必要である. これは, 複数の管理アプリケーション間で管理情報や管理業務の状態を共有し, 更に発生する管理オペレーション及び管理情報アクセスの制御, 同期, 集約を行う.

3　協調管理プラットフォーム

3.1　アプローチ

図 2 に協調管理プラットフォームの概念を示す. 管理アプリケーションが個別に動作している状態(1)から, 排他制御と管理情報の統一, 管理状態の相互参照や引き継ぎが行われる(2)の状態に変化させる.

このような機能は, 各管理アプリケーションの下に層を設けて提供することが効率である. 特に, 標準的な API がない管理アプリケーションを如何に簡単に連携させることができるかが課題となる.

標準的な実現方法としては, 協調管理プラットフォームが管理情報と管理アプリケーションの情報を収集, 管理し, 各管理アプリケーションは管理のロジックを実装する構成がある. 管理アプリケーションは, 協調管理プラットフォームが提供する API を通して管理情報や他の管理アプリケーションの情報にアクセスする. 管理アプリケーションの開発が容易になることが期待できる反面, 以下の問題点がある.

- 既存の管理アプリケーションとの個別対応が必要となり(API ブリッジを開発する等)コストと時間がかかる.
- あらゆる管理アプリケーションで利用する管理情報をプラットフォームで定義するのは大作業となる.
- 既にある管理アプリケーションは管理情報を収集する機能を有しており, 無駄である. また, 特殊な管理アプリケーションでしか使わない管理情報をプラットフォームで持つと, 全体のパフォーマンスに影響が出る.

プラットフォームは, プラグアンドプレイと環境適応性を有することが理想である. 即ち, 管理アプリケーションを動作させるだけで管理情報の共有等が機能し(プラグアンドプレイ), また, それらは, ネットワークトポロジ等に応じて最適化される(環境適応性).

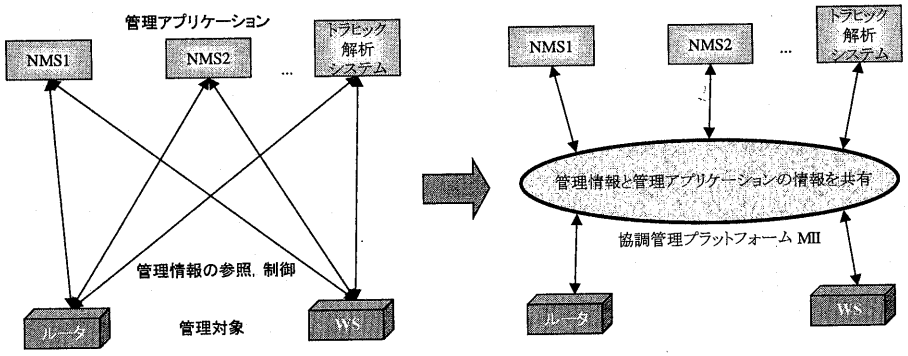
このような協調管理プラットフォームを目指し, 以下のアプローチをとることとする.

- 協調管理プラットフォームは, ある管理アプリケーションが収集した管理情報を, 他の管理アプリケーションが再利用する手段を提供する. 管理アプリケーションの管理パケット(SNMP パケット等)をモニターし管理情報を内部に記憶し, 他の管理アプリケーションから同じ管理情報の取得が起こった際に, 管理対象に代わり返答する. これにより, 管理対象へのアクセスの集約と, 管理情報の共有を, 管理アプリケーションに変更を加えることなく提供することができる.

更に, 管理情報の統一, 管理状態の相互参照を実現する仕掛けを以下の方法で実現する.

- 管理アプリケーションに管理情報の定義と管理状態をアクセスするためのエージェントを準備する(ネットワーク機器の SNMP エージェントとのアナロジー). このエージェントは, 管理アプリケーション毎に異なる管理情報や管理状態の実装方法を隠蔽する.

協調管理プラットフォームは, 管理対象とともに管理アプリケーションの情報を共有する空間となる. 以下, この協調管理



(1) 無秩序に管理アプリケーションを導入した場合　　(2) 協調管理プラットフォームによる連携の実現

図 2　協調管理プラットフォームの概念

プラットフォームを Management Information Interchange (MII)と呼ぶ.

また，協調管理を高度化するには，ヘルプデスク／トラブルチケットシステムなど管理業務の管理を行うシステムが有効である. これらを管理アプリケーションとして同様にプラットフォーム経由で容易に連携させる機構を実現する.

最後のステップとして，排他制御，アクセスコントロールや，管理トラヒックの制御，データの分散配置による処理の効率化や履歴保存等の機能を実現する. 今後新たに開発する管理アプリケーションのため，標準になりつつある分散オブジェクトによるインタフェース(CORBA の IIOP(2)が候補)を設ける.

3.2　利用イメージ

MII を利用した協調管理のイメージを示す(図 3).

- 東京と西海岸にある NOC には，それぞれネットワーク機器が設置されてる. NOC は，両方の機器を昼夜で交代して管理している.
- それぞれの NOC には NMS が設置されている. SNMP トラップは，MII 宛てに送られるよう設定されている.
- 両 NMS とも機器 A～D にポーリングしている. 西海岸にある NMS の A と B に対するポーリングは，東京にある NMS のポーリング結果を MII 経由で利用しており，実際に A や B まで届いていない.

① 今，東京 NOC に新種の機器を追加した. それ用の MIB 定義ファイルを東京 NOC の NMS に設定する.

② MII は，定義ファイルが変更されたことを察知し，協調管理している他の管理アプリケーションに通知.

③ ネットワーク機器は，状態の変化があると，SNMP トラ

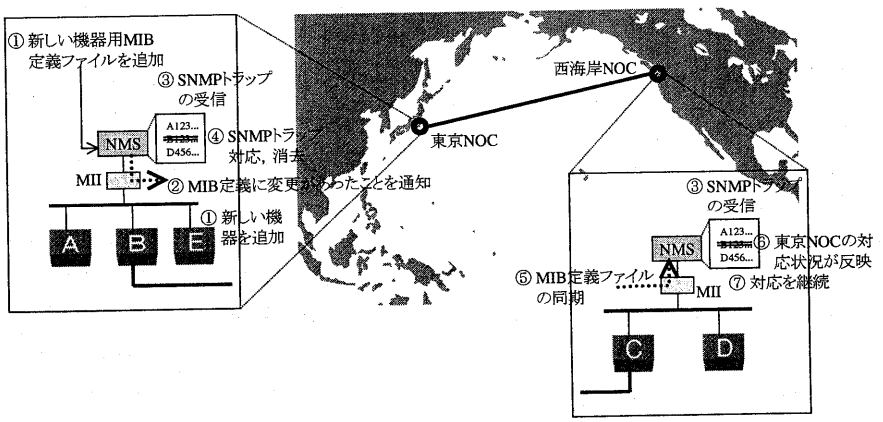


図 3　MII を利用した協調管理の例

ップを MII に送付する. MII は協調管理している管理アプリケーション全てに配布する.

④ 東京 NOC では, NMS が受けた SNMP トラップを見て, 原因を調査, 気にする必要がないことを確認し, トラップを受諾後, 消去する.

⑤ 西海岸NOCで管理する時間になると, 管理者がNMSを見て, MIB ファイルの変更が起こったことを知る. MIB 定義の同期を指示すると, 東京 NOC で追加した機器用の MIB 定義ファイルが西海岸 NMS にも読み込まれる.

⑥ 更に, 西海岸 NOC の NMS において, イベント進捗の同期を指示すると, 東京 NOC で消去や受託した結果がトラップデータに反映される.

⑦ 西海岸 NOC では, その結果残った未解決のトラップやその後来るものについて対処する. 以下繰り返し.

4 実現方法

4.1 管理情報の共有

複数の管理対象と管理アプリケーションを分散配置した場合の管理情報共有機構を実現する. 動作例を 図 4 に示す(管理アプリケーションとして NMS を想定).

管理対象 X, Y はそれぞれ別のロケーションにある. 2 つのロケーションから NMS を用いてこれらを管理している.

① NMS2 が管理対象 Y に対して SNMP の問い合わせを行ったとする. 最初は, MII のデータ収集エージェントはそれを素通りさせる(厳密には, その管理情報が問い合わせ中であることを覚えておく).

② 管理対象から SNMP の回答があると, データ収集エージェントはその内容を見て, 管理情報値をコピーする.

③ コピーした管理情報をデータ管理エージェントに渡し保管する.

④ NMS1 が管理対象 Y に SNMP 問い合わせを発した場合, データ収集エージェントはそのパケットを取る.

⑤ データ管理エージェントに, その管理情報を保持しているか問い合わせる.

⑥ 分散配置されたデータ管理エージェントが協調しながら, 保持している管理情報の値を(ない場合はその旨を)データ収集エージェントに返す.

⑦ データ収集エージェントは管理対象 Y に代わり SNMP で管理情報値を返答する(ない場合は管理対象 Y へ SNMP で問い合わせる).

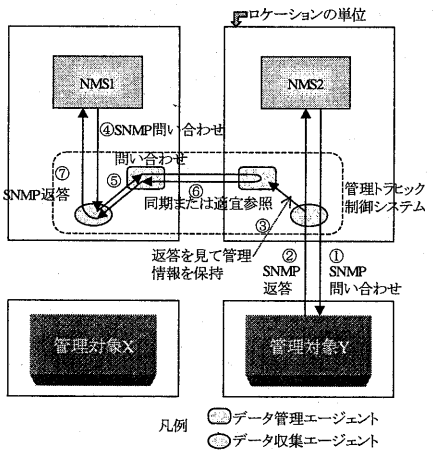


図 4 管理情報共有の動作例

以上の動作例は検討中の幾つかの課題を含んでいる.

(1) 構成

データ収集エージェントを, 管理対象と同じセグメント上に配置すれば, 上記⑥で示した分散処理のオーバーヘッドを減らせる. しかし, プロキシー対応機能のない NMS が存在する場合は, それから出る問い合わせパケットを MII が横取るため, NMS と同じセグメントに配置する必要がある.

(2) データの鮮度

管理アプリケーションや管理情報の種類によって, 鮮度の条件設定をできる機能が必要になる(NMS は管理情報にそれほど厳しい鮮度を要求しない場合が多いが, 管理アプリケーションに同じキャッシュ値を返さない工夫は必要).

(3) データ管理エージェント間の分散データ同期方式

マスターの決定方法, 変更の伝播方法(プッシュ/プルの別)などが課題となる. NMS はポーリングに周期性があるので, それをシステムが学習し, 同期方式を最適化する機能を実現できる可能性がある.

(4) SNMP トラップの配送制御

MII が受けて, 予め登録された管理アプリケーションやスケジュールに沿って転送する. NMS の場合は, ポーリングのトラヒックを監視すれば管理領域を推測できるので, 受けたトラップを, その送付元の機器を管理する NMS 全て(=協調管理している NMS 全て)に配送する適応的な機能が期待できる.

4.2 管理アプリケーション情報の共有

議論してきたように，協調管理では，一般に管理アプリケーション側が保持する以下の情報を共有する必要がある．

- MIB・トラップの意味付け・定義情報
- 管理業務の結果・進捗情報

現在，具体的になっている管理アプリケーションは NMS，管理業務の結果・進捗情報として考えられるのはトラップへの対応状況である．現在，これらの情報は，NMS がローカルに持っているものがほとんどであり，管理アプリケーションがローカルに持っている情報の共有機能を実現しなければならない．MIB やトラップの定義については，ASN.1 という標準記述方法があるが，後者については幾つか解決すべき課題がある．現在までの検討結果を図 5 に示す．

4.1 節で述べたように，MII は全てのトラップを受け取り，適切な NMS に配送する．MII ではその配送記録を保存する(グローバルトラップ DB と呼ぶ)．各 NMS を MII のエージェントが監視し，トラップへの対応状態(読んだ，消去した等)をグローバルトラップ DB に記録する．トラップ対応状況を同期させるリクエストを管理者がした NMS に対して，その NMS がローカルに持つトラップデータの属性を MII が持つトラップ同期規則に基づきエージェントが更新する．トラップ同期規則の例として，一つでも NMS で対応(既読／消去)されたトラップは，他の NMS でも同じ対応状態にする，がある．

残念ながら，これらの機能は，管理アプリケーションの実装方法へ依存する箇所がある(NMS の標準 API or 標準ファイルフォーマットがないため)．3.1 節で述べたプラグアンドプレイを実現するには，動的に管理アプリケーションの種類を判断し，適切なエージェントを選択する機構を実現する必要がある．また，新しい管理アプリケーション用のエージェントの開発量を減らす工夫(オブジェクト指向の継承等を用い，実装依存部分を局所化する)も必要である．

5 おわりに

協調管理プラットフォームの必要性について述べた．個別に管理アプリケーションを導入しても，プラットフォームが自動的に管理トラヒックを集約し，また管理アプリケーション間で管理の状態情報を共有する仕組みを提供する．今後は，プロトタイプシステムを開発し，ISP 等において実用性を評価していく予定である．

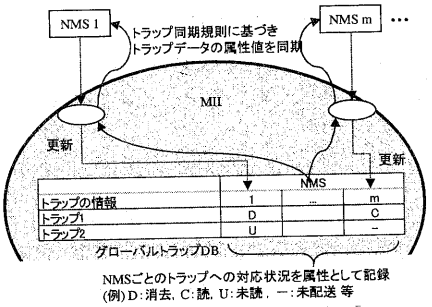


図 5 トラップ対応状況の同期イメージ

参考文献

(1) T. Marshall, "The Simple Book: An Introduction to Internet Management, 2nd Edition," PTR Prentice Hall, Inc., NJ, 1994.

(2) Object Management Group, "The Common Object Request Broker: Architecture and Specification Revision 2.0," OMG Inc., 1995.

(3) M. Hamada, et al., "Building An Application Framework for Computer Network Management Systems," Proc. of the 9th Int'l Conf. on Software Engineering and Knowledge Engineering, Knowledge Systems Institute, IL, 1997, pp. 579-587.

(4) T. Fujisaki, M. Hamada, et al., "A Scaleable Fault-tolerant Network Management System Built Using Distributed Object Technology," Proc. 1st Int'l Enterprise Distributed Object Computing Workshop, IEEE Computer Society Press, 1997, pp. 140-148.

(5) Stallings, W., "Network Management," IEEE Computer Society Press, 1993.

(6) "Using Network Node Manager-HP OpenView Edition 1," Hewlett Packard, 1997.