

情報処理システム管理のためのMIB設計の一考察

4D-8

中安富士夫*¹

濱井和夫*¹

伊藤 弘*² 秋本郁哉*²

*1:NTT 情報システム本部 情報処理技術部 *2:NTTソフトウェア環境情報通信システム事業部

1. 概要

ネットワーク装置のみでなく、広く情報処理システムの管理を行えるようにOSI管理の標準化が進められている。しかし、CMIP/CMIS等のプロトコル/サービスについては標準化されているが、実際に管理を行うオブジェクトの定義で標準化済みのものはネットワーク系に止まっている。

本考察では、メインフレームで構築されたシステムの運転/監視オペレーションにOSI管理を適用した場合、何をオブジェクトとして捉え、どの様にMIBを作成するかについて検討したものである。

2. 管理のモデル

2.1 実現イメージ

メインフレーム系のシステム管理用MIBの標準化が進んでいない状況では、各ベンダホスト上にエージェント（以下、AGT）を実装するのは難しい。そのため、ベンダホストとはベンダ固有メッセージ/コマンドでやり取りを行い、メッセージをEvent, Action/Setをコマンドに変換するプロキシエージェント（以下、P-AGT）を介して図1の様に管理操作を行うことにする。

2.2 エージェントの単位

上記のP-AGT構成を実現するためには、被管理ホスト上にOS, 通信ソフト, 遠隔仮想コンソールパッケージが必要となるため、AGTの最小構

成単位は、OSとなる。また、複数のOSを統合した単位をAGTとして捉える事が可能な構成も考えられるが汎用性を考え、1OS, 1AGTとした。

純粋なハード制御に関しては、複数のCPUセットを統合して制御可能な専用プロセッサを別に設置する構成が多く、ソフトとは別のAGTとした。

図2にAGTの単位を示す。

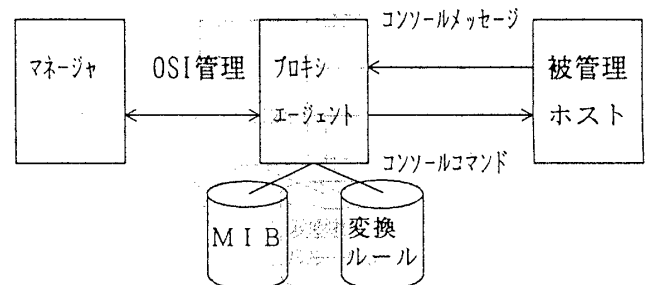


図1. 管理の実現イメージ

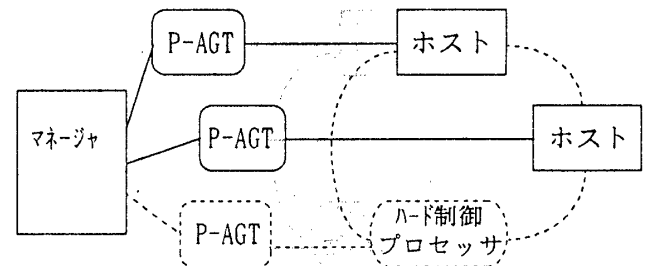


図2. AGTの単位

3. MIB設計

3.1 サービスとサービス資源

情報処理システムを監視する上で重要なのは、「プロセスが障害になった」という事実よりも、そのプロセス障害により「どんなユーザサービス

A Study of MIB-design for information processing system management.

Fujio NAKAYASU*¹, Kazuo HAMAI*¹, Hiroshi ITO*², Ikuya AKIMOTO*²

*1:NTT Information Systems Headquarters Information Processing Technology Department

*2:NTT Software Corporation Communication Processing Systems Sector

が提供できなくなったか」である。

そのため、ソフトの管理においては、ソフトウェアの製品名では無く、「サービス」を管理オブジェクトとし、ソフトの包含木上の最上位を「監視制御サービス」と定義する。「サービス」をオブジェクトにすることにより、各ベンダの差異を吸収することもできる。

また、サービスを提供する上で必要となる論理的な資源（CPU、メモリ、ファイル、データベース等）も管理する必要があり、「サービス資源」と定義する。

3.2 包含関係の考え方

包含木は、管理オブジェクトの物理的、或いは論理的な包含関係を示す。また、この包含木により操作するオブジェクトを一意に決定するため、あるオブジェクトが複数のオブジェクトに包含されることは一般的に禁止される。

そのため、例えばソフトにより仮想的に生成されるオブジェクトと、物理的なオブジェクトは図3の例にあるように包含関係を構成できない。

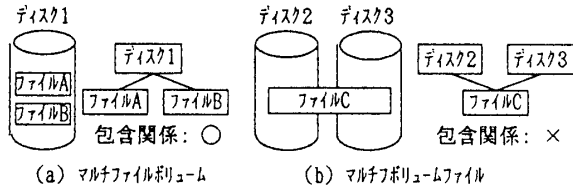


図3. ハード/ソフトの包含関係

従って、ソフトとハードには、包含関係を持ち込まず、ハードはハード、ソフトはソフトの世界での包含関係を構成し、ソフトのサービス資源とハードの物理資源とは図4に示すように関係属性でポイントを張ることとする。

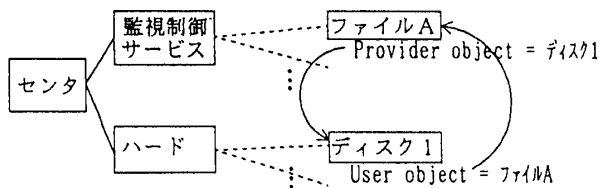


図4. ハード/ソフトの関係属性

サービスとサービス資源の関係も、サービス資源は複数のサービスより使用されるケースが存在する可能性があることより包含関係ではなく、関係属性でポイントを張ることとする。

3.3 ホスト管理MIBの包含木

以上の議論より導かれたシステム管理用MIBの包含木を図5に示す。

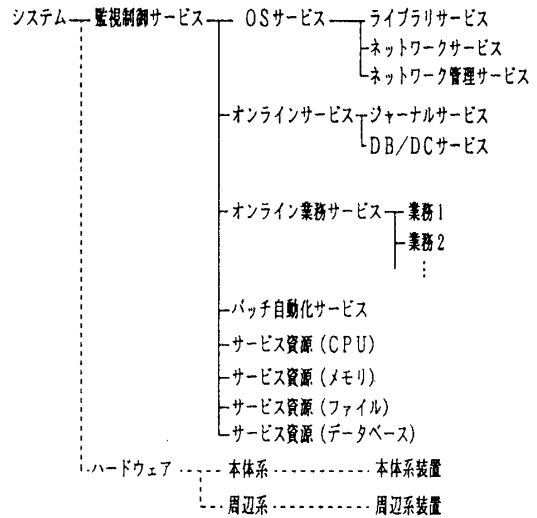


図5. システム管理用包含木

4. 標準とのマッピング

既存のマネージャを有効に使用するには、標準化されたものは、MIBの設計に取り入れた方が良くと考え、適用を行った。クラス自身の定義、アクションの定義は独自のものであるが、属性については、Administrative, Operational, Usageの各Stateによりサービス中、未サービス（異常）、未サービス（正常）のマッピングを行う等の標準化を行った。

イベントについては、State Change, Communication Alarm, Processing Alarm, Reration Change等の標準のものをそのまま使用することとした。

5. 終りに

本考察では、情報処理システムの中のホストに的を絞り、OS I管理を行うためのMIB構成について提案を行った。このMIB構成に基づくプロキシエージェントの設計も並行して行っておりプロキシエージェント方式の妥当性の評価、MIB設計の妥当性検証を今後行っていく予定である。