ＯＳＩ管理に基づくＮＭＳ—ＭＩＢの設計

3Q−2

井崎 智子　和田 哲也　水野 治展　石場 淳

松下電器産業

1．はじめに

ＯＳＩ管理に基づくネットワーク管理システムでは、オブジェクト指向的な設計手法を用いて種々の管理情報を規定する。ＭＩＢ（Management Information Base）は、これらの管理情報を格納するデータベースである。ＭＩＢをリレーショナルＤＢを用いて構築すると、管理情報間の継承関係等の処理が必要となり、管理情報を効率よく操作するのが困難である。

本稿では、管理情報を効率よく操作できるオブジェクト指向ＤＢ型のＭＩＢの実現方法について述べる。

2．基本設計方針

ＭＩＢの基本設計方針は、次のとおりである。

①ＭＩＢはオブジェクト指向ＤＢとし、Ｃ＋＋を用いて構築する。

②ＭＩＢで扱う管理情報の構造は、ＳＭＩ[1]に準拠する。

③ＭＩＢ内部にＳＭＦ[2]を実現するメカニズムをもつ。

④ＭＩＢの保守及び拡張が容易な構造とする。

3．実現方式

3．1 ＭＩＢの構造とＭＩＢインタフェース

ＭＩＢは、図1に示すように複数の管理対象インスタンスをＭＩＴ（Management Informaion Tree）と同一の形状にリンクした構造を持つ。管理対象インスタンスは、ＧＭＯ（Generic Managed Objects）やＳＭＯ（Support Managed Object Classes）で規定される管理対象をＣ＋＋のオブジェクトとして実現したものである。各管理対象インスタンスは、保有している属性タイプに対応した属性インスタンスを複数個もつ。属性インスタンスは、属性タイプ毎の管理情報を保持し、その管理情報に関する操作手段と共にモジュール化されている。管理対象インスタンスは、属性識別子を用いて個々の属性インスタンスをアクセスする。各管理対象インスタンスに対する管理操作は、それぞれが提供する操作関数（Ｃ＋＋のメンバ関数）を呼ぶことによって実現する。これらの操作関数は、全てのインスタンスにおいて共通の関数名をもつ仮想関数として定義する。

ＭＩＢ外部から管理対象インスタンスや属性インスタンスをアクセスする場合のＭＩＢインタフェースを表1に示す。ＭＩＢインタフェースはrootインスタンスの操作関数として実現する。管理アプリケーションは、rootインスタンスの操作関数を呼ぶことによってＭＩＢへの管理操作を実行する。

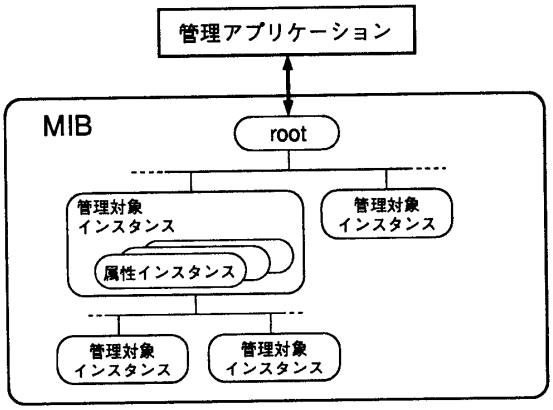

図1．MIBの構成

表1．MIBインタフェース

インタフェース	操作関数
属性操作 if	属性値の読出し 属性値の設定及び置換 属性値の追加 属性値の削除
管理対象操作 if	インスタンスの生成 インスタンスの削除 インスタンスの制御
特権者用操作 if	管理情報の格納 格納された管理情報の読出し 属性値及び内部パラメータの読出し 属性値及び内部パラメータの設定及び置換 属性値及び内部パラメータの追加 属性値及び内部パラメータの削除
層管理 if	層管理からの属性値の変更
イベント通知 if	イベントの報告

3．2 単一インスタンスに対する処理手順

単一の管理対象インスタンス又は属性インスタンスへの管理操作は、それらのＤＮ（Distinguished Name）等を指定してＭＩＢのrootインスタンスの操作関数を呼ぶことによって行う。この管理操作は、rootインスタンス

OSI Management in Network Management System : Design of Management Information Base

Tomoko IZAKI, Tetsuya WADA, Harunobu MIZUNO, Atsushi ISHIBA

MATSUSHITA ELECTRIC INDUSTRIAL CO., LTD.

から目的の管理対象インスタンスに至る経路上に存在する各管理対象インスタンスに対して順次伝達され、目的の管理対象インスタンスに到達した時点で実際の処理が実行される。図２は、単一インスタンスに対する処理手順を示すものである。例えば、ある管理対象インスタンスＸの属性に対して属性値の読出しを行う場合の手順は、次のようになる。

①rootインスタンスに対して属性値の読出し要求を発行する（rootインスタンスのget関数を呼ぶ）。このとき、インスタンスＸのＤＮ等が引数として渡される。

②rootインスタンスからインスタンスＸに至るまでの経路上に存在する各管理対象インスタンスのget関数を順次呼ぶ。get関数の呼出しは、ＭＩＢにおける直上の管理対象インスタンスが行う。

③インスタンスＸのget関数を呼ぶ。

④インスタンスＸのget関数は、アクセスすべき属性インスタンスを検索し、そのget関数を呼ぶ。

⑤属性インスタンスのget関数が属性値の読出しを実行する。

３．３ 複数インスタンスに対する処理手順

管理操作が複数インスタンスにかかわるような場合も、基本的には上記３．２で記述した手順に従って処理する。複数インスタンスに対する処理手順を以下に述べる。

(a)属性インスタンス間の暗黙の関係の処理

ある管理対象インスタンスが保有する複数の属性インスタンス間に、暗黙の関係が存在する場合がある（例. CounterとCounterThreshold）。暗黙の関係をもつ属性インスタンスは、相手の属性型の属性識別子を登録しており、その属性識別子を用いて検索した相手属性インスタンスの操作関数を呼ぶことによって属性インスタンス間の関係を処理する。

(b)関係管理の処理

属性インスタンスＡが関係属性である場合、属性インスタンスＡへの管理操作の影響が、関連する他の属性インスタンスＢにも及ぶ。属性インスタンスＡは、自己宛の管理操作を処理する際に、属性インスタンスＢに対する管理操作を発行する。この管理操作は、属性インスタンスＡからrootインスタンスを経由して属性インスタンスＢに伝える。

(c)イベント通知の処理

イベントは、属性インスタンスの操作時又は管理対象インスタンスの操作時に検出する。検出したイベントの処理手順を以下に述べる（図３参照）。

①イベントが発生した管理対象インスタンスは、イベント情報の編集を行い、そのイベント情報を引数としてrootインスタンスのイベント通知関数を呼ぶ。

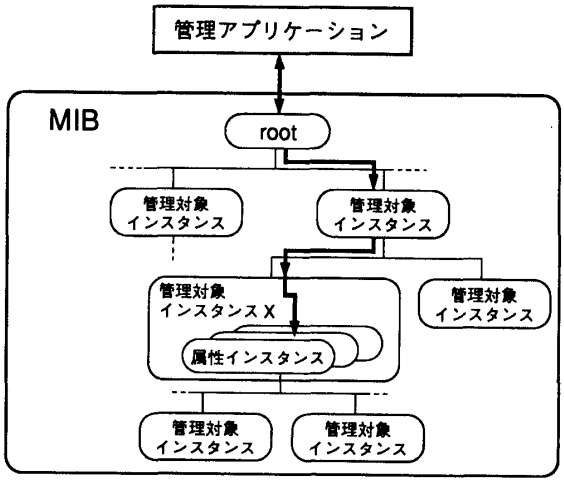

図２．処理の流れ

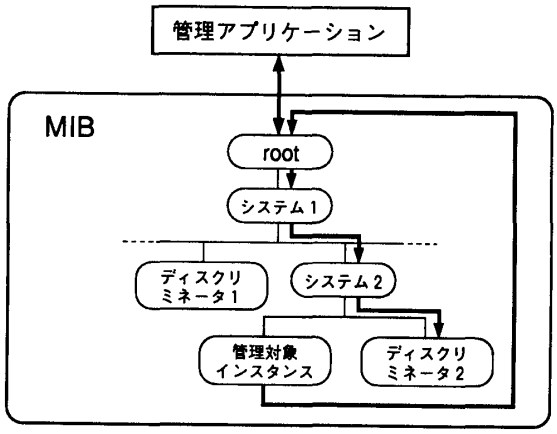

図３．イベントの処理

②ディスクリミネータ２に至る経路上に存在する管理対象インスタンスのイベント通知関数が順次呼ばれる。

③ディスクリミネータ２のイベント通知関数は、イベント情報を解析し、管理アプリケーションへ結果を報告する。

４．あとがき

ＭＩＢをオブジェクト指向ＤＢに基づいて設計する方法について述べた。本稿のＭＩＢでは、各管理情報を同一名の操作関数を備えたＣ＋＋のオブジェクトとして実現し、オブジェクト間をＭＩＴと同一形状によってリンクしているので、管理情報に対する操作を効率よく実現できる。

今後は、本ＭＩＢの内部処理の高速化等について検討を進めていく予定である。

［参考文献］

[1]ISO/IEC DP10165 Part1,2,4(1989) :
OSI-Structure of Management Information
[2]ISO/IEC DP10164 Part1-7(1989) :
OSI-Systems Management Functions