



Solstice Enterprise Agents 1.0 ユー ザーズガイド

Sun Microsystems, Inc.
901 San Antonio Road
Palo Alto, CA 94303
U.S.A. 650-960-1300

Part Number 806-2971-10
2000年3月

Copyright 2000 Sun Microsystems, Inc. 901 San Antonio Road, Palo Alto, California 94303-4900 U.S.A. All rights reserved.

本製品およびそれに関連する文書は著作権法により保護されており、その使用、複製、頒布および逆コンパイルを制限するライセンスのもとにおいて頒布されます。サン・マイクロシステムズ株式会社の書面による事前の許可なく、本製品および関連する文書のいかなる部分も、いかなる方法によっても複製することが禁じられます。

本製品の一部は、カリフォルニア大学からライセンスされている Berkeley BSD システムに基づいていることがあります。UNIX は、X/Open Company, Ltd. が独占的にライセンスしている米国ならびに他の国における登録商標です。フォント技術を含む第三者のソフトウェアは、著作権により保護されており、提供者からライセンスを受けているものです。

RESTRICTED RIGHTS: Use, duplication, or disclosure by the U.S. Government is subject to restrictions of FAR 52.227-14(g)(2)(6/87) and FAR 52.227-19(6/87), or DFAR 252.227-7015(b)(6/95) and DFAR 227.7202-3(a).

本製品に含まれる HG 明朝 L と HG ゴシック B は、株式会社リコーがリコービイマジクス株式会社からライセンス供与されたタイプフェイスマスタをもとに作成されたものです。平成明朝体 W3 は、株式会社リコーが財団法人 日本規格協会 文字フォント開発・普及センターからライセンス供与されたタイプフェイスマスタをもとに作成されたものです。また、HG 明朝 L と HG ゴシック B の補助漢字部分は、平成明朝体 W3 の補助漢字を使用しています。なお、フォントとして無断複製することは禁止されています。

Sun, Sun Microsystems, docs.sun.com, AnswerBook, AnswerBook2 は、米国およびその他の国における米国 Sun Microsystems, Inc. (以下、米国 Sun Microsystems 社とします) の商標もしくは登録商標です。

サンロゴマークおよび Solaris は、米国 Sun Microsystems 社の登録商標です。

すべての SPARC 商標は、米国 SPARC International, Inc. のライセンスを受けて使用している同社の米国およびその他の国における商標または登録商標です。SPARC 商標が付いた製品は、米国 Sun Microsystems 社が開発したアーキテクチャに基づくものです。

OPENLOOK、OpenBoot、JLE は、サン・マイクロシステムズ株式会社の登録商標です。

Wnn は、京都大学、株式会社アステック、オムロン株式会社で共同開発されたソフトウェアです。

Wnn6 は、オムロン株式会社で開発されたソフトウェアです。(Copyright OMRON Co., Ltd. 1999 All Rights Reserved.)

「ATOK」は、株式会社ジャストシステムの登録商標です。

「ATOK8」は株式会社ジャストシステムの著作物であり、「ATOK8」にかかる著作権その他の権利は、すべて株式会社ジャストシステムに帰属します。

「ATOK Server/ATOK12」は、株式会社ジャストシステムの著作物であり、「ATOK Server/ATOK12」にかかる著作権その他の権利は、株式会社ジャストシステムおよび各権利者に帰属します。

本製品に含まれる郵便番号辞書 (7 桁/5 桁) は郵政省が公開したデータを元に制作された物です (一部データの加工を行なっています)。

本製品に含まれるフェイスマーク辞書は、株式会社ビレッジセンターの許諾のもと、同社が発行する『インターネット・パソコン通信フェイスマークガイド'98』に添付のものを使用しています。© 1997 ビレッジセンター

Unicode は、Unicode, Inc. の商標です。

本書で参照されている製品やサービスに関しては、該当する会社または組織に直接お問い合わせください。

OPEN LOOK および Sun Graphical User Interface は、米国 Sun Microsystems 社が自社のユーザおよびライセンス実施権者向けに開発しました。米国 Sun Microsystems 社は、コンピュータ産業用のビジュアルまたはグラフィカル・ユーザインタフェースの概念の研究開発における米国 Xerox 社の先駆者としての成果を認めるものです。米国 Sun Microsystems 社は米国 Xerox 社から Xerox Graphical User Interface の非独占的ライセンスを取得しており、このライセンスは米国 Sun Microsystems 社のライセンス実施権者にも適用されます。

DtComboBox ウィジェットと DtSpinBox ウィジェットのプログラムおよびドキュメントは、Interleaf, Inc. から提供されたものです。(© 1993 Interleaf, Inc.)

本書は、「現状のまま」をベースとして提供され、商品性、特定目的への適合性または第三者の権利の非侵害の黙示の保証を含みそれに限定されない、明示的であるか黙示的であるかを問わない、なんらの保証も行われないものとします。

本製品が、外国為替および外国貿易管理法 (外為法) に定められる戦略物資等 (貨物または役務) に該当する場合、本製品を輸出または日本国外へ持ち出す際には、サン・マイクロシステムズ株式会社の事前の書面による承諾を得ることのほか、外為法および関連法規に基づく輸出手続き、また場合によっては、米国商務省または米国所轄官庁の許可を得ることが必要です。

原典: *Solstice Enterprise Agents 1.0 User Guide*

Part No: 806-2905-10

Revision A



目次

- はじめに 7
- 1. **Solstice Enterprise Agents** とは 11
 - Enterprise Agents の概要 11
 - Enterprise Agents を構成する要素 12
 - SNMP マスターエージェント 13
 - サブエージェント 13
 - エージェントとサブエージェントのソフトウェア開発ツールキット 14
 - レガシー SNMP エージェント 14
 - マッパー 14
 - 用語 14
 - SNMP の機能 15
 - DMI とは 16
 - マスターエージェント 16
- 2. **Enterprise Agents** のインストール 19
 - Enterprise Agents のインストールの概要 19
 - プラットフォームとパッケージ 19
 - SUNWsacom 20
 - SUNWsasnm 20

	SUNWsadmi	20
	SUNWmibii	21
	SUNWsasdk	21
	インストール手順	22
	▼ 既存のパッケージの削除	22
	▼ 新しいパッケージの追加	22
	SNMP のソフトウェアのデフォルトの位置	23
	DMI のソフトウェアのデフォルトの位置	25
3.	SNMP ベースのマスターエージェントとサブエージェント	27
	SNMP ベースのマスターエージェントとサブエージェントの概要	27
	マスターエージェントの起動	27
	サブエージェントの起動	28
	サブエージェントとの通信	29
	サブエージェントの登録	29
	要求の送信	30
	トラップ通知	31
	サブエージェントについて	31
	確立	32
	保守管理	32
	終了	32
	マスターエージェントの使用方法	32
4.	Enterprise Agents の構成	35
	Enterprise Agents の概要	35
	エージェントリソース構成ファイル	35
	environment グループ	37
	resource グループ	37
	エージェント登録ファイル	38
	エージェントアクセス制御ファイル	41

	マスターエージェント状態ファイル	43
	MIB の発行	43
5.	DMI の使用方法	47
	DMI の使用方法の概要	47
	DMTF とは	47
	DMI の機能	48
	DMI のアーキテクチャ	49
	DMI サービスプロバイダ	50
	DMI サービスプロバイダの起動	52
	DMI API ライブラリ	53
	MIF から MIB への変換コンパイラ	53
	マッパー	54
	サブエージェントの初期化と再インストール	55
	マッパーの呼び出し	55
6.	DMI による SNMP の使用	57
	DMI による SNMP の使用の概要	57
	SNMP と DMI との通信	59
	SNMP マスターエージェント	59
	DMI マッパー	59
	SNMP マスターエージェントへの構成要素の登録	59
	Solaris での DMI マッパーの実行	60
	DMI マッパーの機能	61
	▼ サブエージェントの初期化と再初期化	62
	SNMP 要求の DMI への変換	63
	例外の報告	65
	サブエージェントの停止	67
	MIF から MIB へのマップ	67
	特殊なマップの考慮事項	70

	DMI マッパー構成ファイル	71
	WARNING_TIMESTAMP	71
	EXPIRATION_TIMESTAMP	72
	FAILURE_THRESHOLD	72
	TRAP_FORWARD_TO_MAGENT	72
	MIB ファイルの生成	72
7.	DMI コマンド行ユーティリティ	75
	DMI コマンド行ユーティリティについて	75
	dmi_cmd ユーティリティ	75
	dmiget ユーティリティ	76
	dmi_cmd コマンドの使用方法	76
	dmi_cmd の例	78
	dmiget コマンドの使用方法	81
	dmiget の例	82
A.	エラーメッセージ	85
	メッセージの名称	85
	メッセージの数値表現	94
	用語集	101
	索引	113

はじめに

このマニュアルは、Solstice™ Enterprise Agents™ の入門書です。このマニュアルでは、Solstice Enterprise Agents の実行環境と、この製品に関連するサブエージェントのインストール、構成、および管理方法について説明します。

対象読者

このマニュアルは、システム管理者を対象としています。

お読みになる前に

Solstice Enterprise Agents 製品を購入したら、すぐにこのマニュアルをお読みください。製品アーキテクチャの機能、特徴、および構成要素の概要がわかります。

内容の紹介

このマニュアルは、次の章から構成されています。

第 1 章では、製品の概要を説明します。

第 2 章では、Solstice マシンに Solstice Enterprise Agents をインストールする方法について説明します。

第 3 章では、マスターエージェントの機能について説明します。

第 4 章では、構成について説明します。

第 5 章では、デスクトップ管理インタフェースの概要を説明します。

第 6 章では、SNMP の DMI 構成要素へのアクセスの方法と、SNMP と DMI のプロトコル間で通信するときの相互変換の方法を説明します。

第 7 章では、コマンド行インタフェースの機能とその使用例について説明します。

付録 A では、エラーメッセージとその番号のリストを示します。

用語集では、頭字語や略語の意味とともに、このマニュアルで使用している用語の定義について説明します。

関連マニュアル

以下のマニュアルには、このユーザーズガイドに関連する情報が含まれています。

- 『Solaris のシステム管理 (第 1 巻)』

Sun のマニュアルの注文方法

専門書を扱うインターネットの書店 Fatbrain.com から、米国 Sun Microsystems™, Inc. (以降、Sun™ とします) のマニュアルをご注文いただけます。

マニュアルのリストと注文方法については、<http://www1.fatbrain.com/documentation/sun> の Sun Documentation Center をご覧ください。

Sun のオンラインマニュアル

<http://docs.sun.com> では、Sun が提供しているオンラインマニュアルを参照することができます。マニュアルのタイトルや特定の主題などをキーワードとして、検索をおこなうこともできます。

表記上の規則

このマニュアルでは、次のような字体や記号を特別な意味を持つものとして使用します。

表 P-1 表記上の規則

字体または記号	意味	例
AaBbCc123	コマンド名、ファイル名、ディレクトリ名、画面上のコンピュータ出力、コード例を示します。	<code>.login</code> ファイルを編集します。 <code>ls -a</code> を使用してすべてのファイルを表示します。 <code>system%</code>
AaBbCc123	ユーザーが入力する文字を、画面上のコンピュータ出力と区別して示します。	<code>system% su</code> <code>password:</code>
AaBbCc123	変数を示します。実際に使用する特定の名称または値で置き換えます。	ファイルを削除するには、 <code>rm filename</code> と入力します。
『 』	参照する書名を示します。	『コードマネージャ・ユーザーズガイド』を参照してください。
「 」	参照する章、節、ボタンやメニュー名、強調する単語を示します。	第 5 章「衝突の回避」を参照してください。 この操作ができるのは、「スーパーユーザー」だけです。
\	枠で囲まれたコード例で、テキストがページ行幅を超える場合に、継続を示します。	<code>sun% grep `^#define` \</code> <code>XV_VERSION_STRING'</code>

ただし AnswerBook2™ では、ユーザーが入力する文字と画面上のコンピュータ出力は区別して表示されません。

コード例は次のように表示されます。

■ C シェルプロンプト

```
system% command y|n [filename]
```

- Bourne シェルおよび Korn シェルのプロンプト

```
system$ command y|n [filename]
```

- スーパーユーザーのプロンプト

```
system# command y|n [filename]
```

[] は省略可能な項目を示します。上記の例は、*filename* は省略してもよいことを示しています。

| は区切り文字 (セパレータ) です。この文字で分割されている引数のうち 1 つだけを指定します。

キーボードのキー名は英文で、頭文字を大文字で示します (例: Shift キーを押します)。ただし、キーボードによっては Enter キーが Return キーの動作をします。

ダッシュ (-) は 2 つのキーを同時に押すことを示します。たとえば、Ctrl-D は Control キーを押したまま D キーを押すことを意味します。

一般規則

- このマニュアルでは、英語環境での画面イメージを使っています。このため、実際に日本語環境で表示される画面イメージとこのマニュアルで使っている画面イメージが異なる場合があります。本文中で画面イメージを説明する場合には、日本語のメニュー、ボタン名などの項目名と英語の項目名が、適宜併記されています。
- このマニュアルでは、「IA」という用語は、Intel 32 ビットのプロセッサアーキテクチャを意味します。これには、Pentium、Pentium Pro、Pentium II、Pentium II Xeon、Celeron、Pentium III、Pentium III Xeon の各プロセッサ、および AMD、Cyrix が提供する互換マイクロプロセッサチップが含まれます。

Solstice Enterprise Agents とは

- 11ページの「Enterprise Agents の概要」
- 12ページの「Enterprise Agents を構成する要素」
- 15ページの「SNMP の機能」
- 16ページの「DMI とは」
- 16ページの「マスターエージェント」

Enterprise Agents の概要

簡易ネットワーク管理プロトコル (SNMP) は、システム、ネットワークを構成する装置、およびネットワークそのものを効率的に管理するために、企業ネットワークで幅広く使われています。SNMP の普及に伴い、システムやネットワークの管理に関連する多くの問題点も出てきました。SNMP の利点の 1 つとして、増加しつつあるネットワーク構成要素やアプリケーションに対応するためのソリューションを迅速に提供できることが挙げられます。

SNMP ネットワーク内で、管理を必要とするエンティティ (システム、構成要素、およびアプリケーション) の数は急速に増えています。そこで、複数の装置をより柔軟に、しかも動的に管理したいという要求に応える必要があります。

初期の SNMP ベースのネットワーク管理ソリューションでは、開発者は、システムまたは装置ごとに単位型エージェントを 1 つずつ作成していました。この単位型エージェントは、単独のポート (ポート 161) で受信処理を行います。しかし、こ

の SNMP のソリューションには多くの制約があったため、必要なすべての装置を効率的に管理するという点で柔軟性に欠けていました。

1 つの装置で複数の構成要素やアプリケーションを個別に管理できるようにするため、複数の開発者が複数のエージェントを作成するための技術が求められました。その結果、新しい拡張性のあるエージェント技術、つまりマスター / サブエージェント技術が生まれました。サンでは、この技術に基づいた **Solstice Enterprise Agents (SEA)** というソリューションを提供しています。

この場合エージェントは、マスターエージェントとサブエージェントで構成されます。マスターエージェントは、マネージャから SNMP ベースの管理要求を受信し、これらの管理要求に対する応答を返送します。それぞれのサブエージェントから適切な値を受信してから、応答が返送されます。サブエージェントは、構成要素またはアプリケーション用に特別に設計された管理情報ベース (MIB または MIF) に従って、さまざまな構成要素を管理します。

また、Enterprise Agents では、SNMP ベースのレガシーエージェントを統合して使用できます。

マスターエージェントとサブエージェントの役割については、あとの章で詳しく説明します。

Enterprise Agents を構成する要素

SNMP ベースの SEA 製品は、さまざまな要素から構成されています。

図 1-1 に、SEA のアーキテクチャを示します。

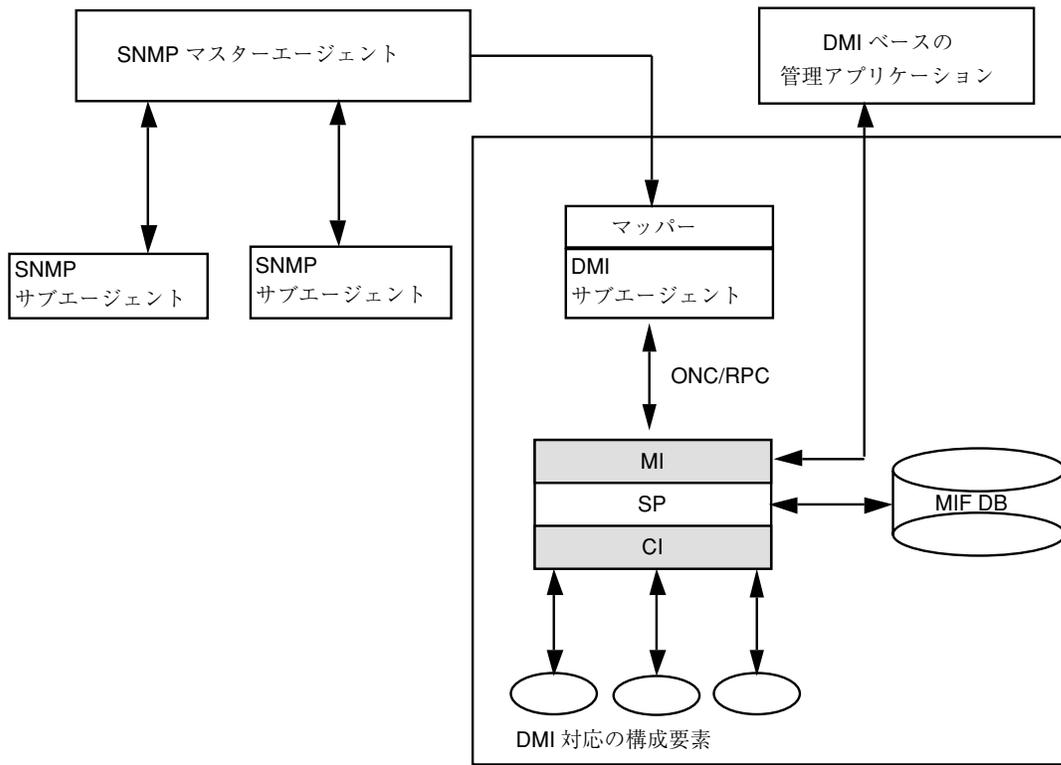


図 1-1 SEA のアーキテクチャ

次に、SEA 製品に関連する各構成要素について説明します。

SNMP マスターエージェント

マスターエージェントは、Domain Manager™、Enterprise Manager™、HP OpenView などの各マネージャと SNMP のプロトコルメッセージを交換する、被管理ノード上のエンティティまたはプロセスです。

マスターエージェントは、ポート 161 で受信します。

サブエージェント

サブエージェントは、管理情報にアクセスできる 0 (ゼロ) 個以上のプロセスです。これによってシステム内のさまざまなアプリケーションや構成要素に対する管理機

能を發揮します。サブエージェントは、SNMP でマスターエージェントと通信します。サブエージェントがマネージャに直接アクセスすることはありません。

エージェントとサブエージェントのソフトウェア開発ツールキット

ソフトウェア開発ツールキットには、複数の構成要素があります。このツールキットには、エージェント / サブエージェントライブラリ、MIB コンパイラ、およびサブエージェントのサンプルが含まれています。MIB コンパイラは MIB を解析し、スタブファイルを作成します。スタブファイルは、必要に応じて修正したり拡張できる機能を収めたものです。これによって、それぞれの構成要素やアプリケーションを管理できます。

レガシー SNMP エージェント

レガシー SNMP エージェントは、SNMP ベースのエージェントであり、システム内で単位型エンティティとして機能します。Enterprise Agents を使うと、レガシー SNMP エージェントを統合できます。レガシーエージェントとは、サン・マイクロシステムズ社または他の会社からすでにリリースされている製品のエージェントのことです。

マッパー

Enterprise Agents の技術を使用することにより、DMI 2.0 の機能も統合できます。これは、サブエージェントとしての役割を果たすマッパーによって行われます。マッパーは、マスターエージェントから要求を受信すると、それらを適切な DMI 要求に変換します。変換された DMI 要求は、DMI サービスプロバイダに送信されます。マッパーは、DMI サービスプロバイダからの応答を受信すると、その応答を SNMP 応答に変換してから、マスターエージェントを通じてマネージャに転送します。

用語

SEA 製品を使う際には、次に示す用語を知っておく必要があります。

「登録」は、マスターエージェントがサブエージェントから情報を受信すると行われる処理です。登録後に、サブエージェントに MIB サブツリーの管理が提供されます。

「サブツリー」は、1つの OID によって示されます。マスターエージェントは、MIB 定義にないサブツリーは認識しません。サブツリーは、実際には「host」のように MIB 全体であったり、「hrDeviceDescr.42」のようにフルインスタンスであったりします。また、MIB 定義で名前が指定されていないサブツリーもあります。

「OID」の範囲は、サブツリーに組み込まれている OID の範囲です。たとえば、サブツリー 1.2.3 は、1.2.3 を含んだそれ以降の範囲を表しますが、1.2.4 は含まれません。

「二重登録」は、あるサブエージェントによってすでに登録されているサブツリーと同一のサブツリーを、別のエージェントが登録しようとすることです。

「重複登録」は、あるサブエージェントによってすでに登録されているサブツリーに含まれているか、またはそのサブツリーを含んでいるサブツリーを、別のエージェントが登録しようとすることです。

「ディスパッチ」は、マスターエージェントから1つ以上のサブエージェントに管理要求を送信することです。ディスパッチは、マスターエージェントの登録されたサブツリーの現在の状態と、明示的に定義されたアルゴリズムに従って行われます。

そのほかの用語については、このマニュアルの用語集で説明します。

SNMP の機能

マスターエージェントは、システムマネージャから SNMP 要求を受信すると、サブエージェントから適切な値を受けとってこれらの要求に対する応答を返送します。サブエージェントは、構成要素やアプリケーション用に特別に設計された管理情報ベース (MIB) に従って、さまざまな構成要素を管理します。動的に呼び出されたサブエージェントは、それぞれマスターエージェントに登録されます。この登録中に、サブエージェントが管理している MIB サブツリーをマスターエージェントに通知します。詳細は、第3章を参照してください。

SEA 技術は、サブエージェントを作成、リリース、およびインストールできるソフトウェア開発キットを提供します。また、SEA を使えば、SNMP ベースのレガシーエージェントを統合して使えます。

DMI とは

デスクトップ管理インタフェース (DMI) は、システム内に存在する管理アプリケーションと構成要素の間を仲介する、インタフェースとサービスプロバイダです。DMI は、特定のオペレーティングシステムや管理プロセスに結び付かない独立したインタフェースです。

サンでは、サンのプラットフォームおよびそのプラットフォームで動作するソフトウェアアプリケーションを管理するための DMI ベースの機能を提供しています。ここでいうサンのプラットフォームには、ハードウェアとソフトウェアが含まれます。DMI サブエージェントは、SEA 製品に含まれるサブエージェントの 1 つです。DMI を使うことによって、ほとんどのシステム内のさまざまな要素を管理できます。これらの要素には、PC、ワークステーション、ルーター、ハブ、その他のネットワークオブジェクトなどがあります。

DMI は、次の 4 つの要素で構成されています。

- 管理情報を記述するための書式 (MIF)
- サービスプロバイダのエンティティ
- 2 種類の API
 - 管理アプリケーションとサービスプロバイダとのインタフェース
 - サービスプロバイダとコンポーネントインストールメンテーションとのインタフェース
- リモート通信を容易にするためのサービス (ONC™ / PRC を使用)

詳細は、第 5 章を参照してください。

マスターエージェント

マスターエージェントは、ネットワークマネージャとサブエージェントの間の主インタフェースとしての役割を果たします。ネットワークマネージャから送信された要求は、マスターエージェントによって解析されます。必要に応じて、元の要求は複数の要求に分割されることがあります。元の要求は、各サブエージェントが提供する管理特性に基づき、マスターエージェントから配信されます。要求は適切なサブエージェントに転送され、各要求に対する応答が返送されます。それぞれのサ

ブエージェントからすべての応答を収集すると、最終の応答がネットワークマネージャに送信されます。

1つのマスターエージェントが、マスターエージェント / サブエージェントモデル全体を管理します。マスターエージェントは、すべての登録されたサブエージェントに対する要求のスケジューラとディスパッチャとして機能します。また、サブエージェントがマスターエージェントにトラップを送信すると、そのトラップはネットワークマネージャに転送されます。

図 1-2 に、SEA アーキテクチャに関連するマスターエージェントを示します。

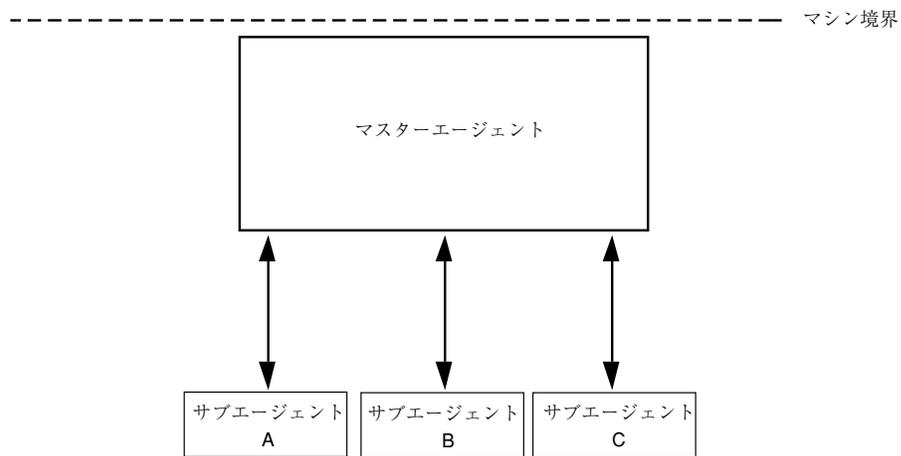


図 1-2 マスターエージェントのアーキテクチャ

Enterprise Agents のインストール

- 19ページの「Enterprise Agents のインストールの概要」
- 19ページの「プラットフォームとパッケージ」
- 22ページの「インストール手順」
- 23ページの「SNMP のソフトウェアのデフォルトの位置」
- 25ページの「DMI のソフトウェアのデフォルトの位置」

Enterprise Agents のインストールの概要

Solaris™ オペレーティングシステムに Solstice Enterprise Agents (SEA) をインストールするには、以降の節に示す Solaris の標準規約に従います。

SEA パッケージは、主に次の 2 つの機能に分かれています。

- SNMP
- DMI

プラットフォームとパッケージ

この製品は、次のプラットフォームに対応しています。

- Sparc (Solaris 2.4、2.5、2.5.1、2.6、Solaris 7)

- IA (Solaris 2.4、2.5、2.5.1、2.6、Solaris 7)

実行時製品には、次のような固有のパッケージがあります。

- SUNWsacom
- SUNWsasnm
- SUNWsadmi
- SUNWmibii
- SUNWsdsk (エージェント開発用のオプションパッケージ)

以降の節で、それぞれのパッケージについて詳しく説明します。

SUNWsacom

SUNWsacom パッケージには、ほかの3つのパッケージに対応する構成ファイルがすべて含まれています。このパッケージ内のファイルは、/ (ルート) ファイルシステムと /var ファイルシステムにインストールされます。これらのファイルシステムには、構成ファイルと共有ファイルが含まれています。

SUNWsasnm

SUNWsasnm パッケージには、次のファイルが含まれています。

- snmpdx - マスターエージェントの実行形式ファイル
- init.snmpdx - スタートアップスクリプトファイル
- snmpdx.mib - マスターエージェント MIB ファイル
- ライブラリ

SUNWsadmi

SUNWsadmi パッケージには、次のファイルが含まれています。

- dmispd - DMI サービスプロバイダ (SP) の実行形式ファイル
- DMI ライブラリ - 表 2-3 を参照してください。
- snmpXdmid - DMI / SNMP マッパーの実行形式ファイル
- init.dmi - スタートアップスクリプトファイル

- `ciinvoke - init.dmi` から CI エージェントを呼び出すためのスクリプト
- CI エージェントを呼び出すためのスクリプト - `ciinvoke` によって呼び出される
- `dmi_cmd` と `dmiget - コマンド行 MI ユーティリティ`

SUNWmibii

SUNWmibii パッケージには、`mibiisa` MIB II サブエージェントが含まれています。このサブエージェントは、MIB II の機能を提供します。MIB II サブエージェントによって提供されるこの機能は、Domain Manager でリリースされる `snmpd` エージェントによって提供される機能と同じものです。

SUNWsasdk

このパッケージには、サブエージェントを構築するための SNMP ツールキットと DMI ツールキットが含まれています。

SNMP ツールキットには、次のものが含まれています。

- `mibcodegen`
- ヘッダーファイル
- 標準 MIB ファイル
- サンプル MIB とサンプルコード

DMI ツールキットには、次のものが含まれています。

- `miftomib` コンパイラ
- `libdmi`
- `libci`
- `libmi`
- `sp.mif`
- CI および MI のサンプルファイル

インストール手順

パッケージをインストールするときは、`pkgadd` コマンドを使います。また、パッケージを削除するときは、`pkgrm` コマンドを使います。

注 - `snmpXdmid` は、マスターエージェントの構成ファイルで正しく設定する必要があります。

▼ 既存のパッケージの削除



注意 - インストールの手順を開始する前に、必ずスーパーユーザーとしてログインしてください。それから、下記の手順に従います。

既存のパッケージがなければ、削除する必要はありません。

▼ 新しいパッケージの追加

パッケージをそれぞれ追加すると、英語とフランス語の両方の著作権が表示され、続いて一連のプロンプトが表示されます。このとき、特定の応答を入力するか、またはデフォルトを受け入れます。

1. **SUNWmibii** の追加

```
pkgadd -d . SUNWmibii
```

2. **SUNWsasnm** の追加

```
pkgadd -d . SUNWsasnm
```

3. **SUNWsadmi** の追加

```
pkgadd -d . SUNWsadmi
```

4. **SUNWsacom** の追加

```
pkgadd -d . SUNWsacom
```

5. **SUNWsasdk** の追加

```
pkgadd -d . SUNWsasdk
```

SUNWsadmi パッケージを追加すると、システムの再起動の直後に dmispd プロセスと snmpXdmid プロセスが開始されます。

SUNWsacom パッケージは、init.dmi と init.snmpdx の 2 つのスクリプトファイルで構成されています。SNMP のデーモン snmpdx は、init.snmpdx によって呼び出されます。snmpdx プロセスは、自動的に mibiisa デーモンを起動します。

dmispd プロセスは、システムがブートされると、snmpdx と同様に RC スクリプトファイルを通じて呼び出されます。snmpXdmid マッパープロセスは、dmispd に続いて呼び出されます。

注 - パッケージを削除したり追加したりするときは、必ず上記の順序に従ってください。

SNMP のソフトウェアのデフォルトの位置

表 2-1 に、SNMP のソフトウェアのデフォルトの位置を示します。

表 2-1 デフォルトの SNMP の位置

ラベル	ディレクトリ
SEA_SNMPLibrary	/usr/lib
SEA_SNMPConfiguration_Directory	/etc/snmp/conf
SEA_SNMPMibs_Directory	/var/snmp/mib

表 2-2 に、SNMP の構成要素の名前とそのデフォルトの位置を示します。

表 2-2 SNMP パッケージの構成要素

構成要素の名前	ラベルまたはディレクトリ	説明
snmpdx	/usr/lib/snmp	マスターエージェントの実行形式ファイル
mibiisa	/usr/lib/snmp	MIB II snmp daemonpwd
snmp_trapshnd	/usr/sbin	トラップ送信用ユーティリティ
mibcodegen	/usr/bin	コードジェネレータの実行形式ファイル
snmpdx.rsrc	SEA_SNMPConfiguration_Directory	マスターエージェントのリソースファイル
snmpdx.reg	SEA_SNMPConfiguration_Directory	エージェントの登録ファイル
snmpdx.acl	SEA_SNMPConfiguration_Directory	マスターエージェントのアクセス制御ファイル
snmpd.conf	SEA_SNMPConfiguration_Directory	SNMPD 構成ファイル
mibiisa.reg	SEA_SNMPConfiguration_Directory	MIB II サブエージェントの登録ファイル
mibiisa.rsrc	SEA_SNMPConfiguration_Directory	MIB II エージェントのリソースファイル
snmpdx.st	SEA_SNMPRun_Time_Directory	マスターエージェントの状態ファイル
libssasnmplib.so.1	SEA_SNMPLibrary_Directory	SSA SDK SNMP ライブラリ
libssagent.so.1	SEA_SNMPLibrary_Directory	SSA SDK Agent ライブラリ
enterprises.oid	SEA_SNMPConfiguration_Directory	デフォルトのエンタープライズ名の OID マップ
sun.mib	SEA_SNMPMibs_Directory	Sun MIB
snmpdx.mib	SEA_SNMPMibs_Directory	Snmpdx MIB

DMI のソフトウェアのデフォルトの位置

表 2-3 に、製品の DMI に関するソフトウェアのデフォルトの位置を示します。

表 2-3 DMI に関するデフォルトの位置

ラベル	位置
SEA_DMILibrary_Directory	/usr/lib
SEA_DMIConfiguration_Directory	/etc/dmi/conf
SEA_DMIRunTime_Database_Directory	/var/dmi/db
SEA_DMIRunTime_MAP_Directory	/var/dmi/map
SEA_DMIMif_Directory	/var/dmi/mif

表 2-4 に、DMI の構成要素の名前とそのデフォルトの位置を示します。

表 2-4 DMI パッケージの構成要素

構成要素の名前	ラベルまたはディレクトリ	説明
snmpXdmid	/usr/lib/dmi	マッパーの実行形式ファイル
dmispd	/usr/lib/dmi	DMI サービスプロバイダの実行形式なファイル
dmi_cmd	/usr/sbin	DMI コマンドユーティリティ
dmiget	/usr/sbin	DMI コマンドユーティリティ
snmpXdmid.conf	SEA_DMIConfiguration_Directory	マッパーの構成ファイル
dmispd.conf	SEA_DMIConfiguration_Directory	DMI SP 構成ファイル
map files	SEA_DMIRunTime_MAP_Directory	マップファイル

表 2-4 DMI パッケージの構成要素 続く

構成要素の名前	ラベルまたはディレクトリ	説明
libdmi.so.1	SEA_DMILibrary_Directory	SSA SDK DMI 汎用ライブラリ
libci.so.1	SEA_DMILibrary_Directory	SSA SDK CI ライブラリ
libdmimi.so.1	SEA_DMILibrary_Directory	SSA SDK MI ライブラリ
sp.mif	SEA_DMIMif_Directory	MIF ファイル
ciinvoke	/etc/dmi/ciagent	DMI 構成要素のインタフェース起動スクリプト

SNMP ベースのマスターエージェントとサブエージェント

- 27ページの「SNMP ベースのマスターエージェントとサブエージェントの概要」
- 31ページの「サブエージェントについて」
- 32ページの「マスターエージェントの使用方法」

SNMP ベースのマスターエージェントとサブエージェントの概要

マスターエージェントは、Solstice Enterprise Agents 技術の主要構成要素です。マスターエージェントは、デーモンプロセスとして動作し、ユーザーデータグラムプロトコル (UDP) のポート 161 で SNMP 要求を受信します。またマスターエージェントは、ほかのポートをオープンして、さまざまなサブエージェントから SNMP トラップ通知を受信します。これらのトラップは、構成ファイルでの定義に従ってさまざまなマネージャに転送されます。

マスターエージェントの起動

システムを最初に起動すると、システムの起動スクリプトファイルによってマスターエージェントが呼び出されます。マスターエージェントが起動されると、さまざまな構成ファイルが読み込まれます。次に、サブエージェントが動作可能にな

り、さまざまなサブエージェントのサブツリー OID を定義し、サブエージェント自身の MIB を設定することによって、適切なアクションが実行されます。マスターエージェントは、次の機能を提供します。

- サブエージェントの起動
- サブエージェントとの通信
- サブエージェントの登録
- サブエージェントへの要求送信
- サブエージェントからの応答受信
- サブエージェントからの通知トラップ

注 - ポート 161 を使用する snmpd (Domain Manager の一部) などの SNMP エージェントを実行している場合は、Solstice Enterprise Agents は実行できません。

サブエージェントの起動

サブエージェントは、次の方法で起動できます。

- マスターエージェントにより起動する方法 - マスターエージェントは、SEA_SNMPConfiguration_Directory にあるエージェントリソースファイルを使ってすべてのエージェントを起動できます。エージェントは、マスターエージェントのリソースファイルの指定に従って起動されます。マスターエージェントがサブエージェントの起動に成功すると、そのサブエージェントのテーブルのエントリ行が作成され、そのサブエージェントに対する適切な値が、MIB 変数に代入されます。次に、サブエージェントがマスターエージェントに登録され、マスターエージェントから SNMP 要求を受信できるようになります。
- システムブート時に手動または自動で起動する方法 - システム管理者やユーザーは、エージェントリソースファイルがないサブエージェントを手動で起動できます。あるいは、システムの起動時に、起動スクリプトによってサブエージェントを起動することもできます。これらのエージェントは、マスターエージェントを起動したあとにしか起動できません。また、これらのエージェントは、Solstice Enterprise Agents のソフトウェア開発キット (SDK) を使って作成し、適切なライブラリにリンクしておく必要があります。それによって、サブエージェントをマスターエージェントに動的に登録できます。

サブエージェントとの通信

サブエージェントからマスターエージェントへの通信は、UDP ポート 161 を使って行われます。サブエージェントからマスターエージェントへのトラップの送信については、31ページの「トラップ通知」で説明します。

注・マスターエージェントは、サブエージェントごとに個別のポートで通信を行います。

また、マスターエージェントは、登録されているサブエージェントが起動し、動作しているかどうかを、次の条件をもとに確認します。

- マスターエージェントによって各サブエージェントに送信される、Get、Get Next、および Set 要求ごとのタイムアウト機構
- マスターエージェントとサブエージェントとの間にアクティビティが存在しないこと。マスターエージェントは、エージェントリソースファイルで定義されている `watch_dog_time` に従って、特定のサブエージェントが実行状態かどうかを判断します。ある特定の設定期間にサブエージェントとマスターエージェントとの間にアクティビティがない場合は、SNMP Get 要求をサブエージェントに送信することによってサブエージェントが動作可能になります。

サブエージェントの登録

サブエージェントを登録するために、マスターエージェントは、サブエージェントを MIB に結び付けます。次に、マスターエージェントは、以下のどちらかの方法を使って、サブエージェントの現在の位置を決めます。

- 静的な方法 - マスターエージェントは、エージェントリソースファイルを読み込みます。このリソースファイルには、それぞれのサブエージェントのエントリが含まれています。
- 動的な方法 - マスターエージェントは、サブエージェントから情報を受信します。サブエージェントは、登録 API を使って、マスターエージェントに登録するために必要な MIB オブジェクトを含む SET 要求を送信します。

バインドポリシーは、SNMP オブジェクト識別子 (OID) の登録に関するものです。これは、さまざまなサブエージェントに SNMP 要求をディスパッチする際のマスターエージェントの意思決定に影響します。マスターエージェントは、表 3-1 に示すバインドポリシーをサポートします。

表 3-1 バインドポリシー

登録の種類	登録方法
個別変数登録	サブエージェントは個々の変数を管理できる
行の登録	サブエージェントは各行または複数の行を管理できる
テーブルの登録	サブエージェントは、テーブルの全体および一部分に登録を行うことができる。テーブルの一部分の登録とは、テーブルのいくつかのカラムを登録することである。たとえば、テーブルに c1~c5 のカラムがある場合、サブエージェントは、そのテーブルの c3 と c5 のカラムだけを登録できる
二重登録	二重登録は行えない
重複登録	重複登録の場合、マスターエージェントは、OID が最も一致している要求をディスパッチする

要求の送信

マスターエージェントは、2つのモードによる SNMP 要求 (Get、Get Next、および Set) の転送をサポートしています。省略可能な引数をコマンド行呼び出しに指定することによって、モードが示されます。モードは次のとおりです

- グループモード – マスターエージェントからサブエージェントへのそれぞれの要求に複数の変数を含めることができます。したがって、エージェントごとに1つの送信要求になります。
- 分割モード – 受信する要求の各変数が、それぞれのサブエージェントに対する1つの送信要求になります。

送信可能な要求を次に示します。

- GET および GETNEXT – 要求をサブエージェントに送ります。
- SET – マルチフェーズ形式で設定を実装します。まず、Set 要求内のすべての varbind が、Get 要求によって検索されます。Get 要求が成功すると、Set 要求が各サブエージェントに送信されます。Set 要求が成功すると、SUCCESS 応答がマネージャに送信されます。Set 要求が失敗すると、別の Set 要求が元の値とともにサブエージェントに送信され、失敗した Set 要求は取り消されます。Set 要求は、処理中の他の SNMP 要求が完了してから開始されます。

- GET RESPONSE – サブエージェントから応答を受信します。
- TRAP – サブエージェントから通知を受信します。

トラップ通知

サブエージェントは、マスターエージェントにトラップを送信します。マスターエージェントは、どのマネージャがトラップを受信するかを決めます。この判断は設定可能です。

サブエージェントについて

サブエージェントは、ネットワークマネージャと直接通信することはありません。その代わりに、マスターエージェントと通信します。サブエージェントの管理責任はマスターエージェントに委ねることができます。ただし、それを受諾するか拒否するかはマスターエージェントによって判断されます。

サブエージェントは、マスターエージェントに登録されてから、マスターエージェントからの SNMP 要求を待ちます。要求を受信すると、サブエージェントは適切な応答を返送します。さらに、サブエージェントは、SNMP トラップを送信することもできます。

サブエージェントは、4つの主要要素で構成されます。その4つの要素とは、SNMP トラップ、サービス API スタック、サブエージェントアプリケーション、MIB データベースです。マスターエージェントは、SNMP のプロトコルデータユニット (PDU) の送受信を管理します。また、PDU のコード化と復号化を行います。SP (スタック) は、受け取った要求を適切にディスパッチしたり、適用可能なコールバック関数を呼び出します。

システムインタフェースモジュールは、サブエージェントによって管理されているすべての変数に対するコールバックを実装したものです。MIB コンパイラは、自動的にこの情報を生成します。

システムに依存するインタフェースと MIB データベース以外の他の構成要素は、他のサブエージェントによって再利用可能です。これらの構成要素は、ライブラリで提供されています。

確立

サブエージェントは、サブエージェント自身の存在を知らせ、通信方法を指定します。トランスポートは、UDP が使用されます。サブエージェントが SNMP 要求を受信するポートは設定可能です。詳細については、41ページの「エージェントアクセス制御ファイル」を参照してください。ハンドシェイクプロトコルを使用して動的にこの確立処理を行うこともできます。

保守管理

サブエージェントは、マスターエージェントの存在を定期的を確認します。マスターエージェントによって呼び出されていない動的サブエージェントは、マスターエージェントが実行状態であるかどうかを定期的に判断します。実行時ライブラリは、割り込みイベントによって、サブエージェントのサブツリーをマスターエージェントに登録します。

終了

終了時には、動的サブエージェントが終了しようとしていることをマスターエージェントに知らせます。そのあと、マスターエージェントは、メモリー内のサブエージェントのテーブルからその行のエントリを削除できます。

マスターエージェントの使用方法

```
snmpdxdx [-h] [-p port_number] [-r filename]  
[-a filename] [-c dirname] [-i filename] [-o filename] [-y]  
[-m GROUP|SPLIT] [-d debug_level]
```

コマンド行の引数を、表 3-2 に示します。

表 3-2 マスターエージェントのコマンド行引数

引数	説明
-a <i>filename</i>	アクセス制御ファイルのフルパスを指定する。デフォルトのファイルは、 <code>/etc/snmp/conf/snmdx.acl</code> 。詳しくは、41ページの「エージェントアクセス制御ファイル」を参照
-c <i>dirname</i>	エージェントのリソースファイルが含まれているディレクトリのフルパスを指定する。 デフォルトのディレクトリは、 <code>/etc/snmp/conf</code>
-d <i>debug_level</i>	デバッグのために使用される。 <i>debug_level</i> (0~4) に応じて、特定の情報を印刷する。デフォルトの <i>debug_level</i> は 0
-h	コマンドの使用方法を表示
-i <i>filename</i>	クラッシュ後の復旧の際にマスターエージェントによって使用される PID のフルパスを指定する。UNIX プロセス ID、ポート番号、リソース名、およびエージェント名などが含まれる。デフォルトのファイルは、 <code>/var/snmp/snmpdx.st</code>
-o <i>filename</i>	このファイルには、(Sun Microsystems, 1.3.1.6.1.4.32) のように (<i>enterprise_name, oid</i>) の組み合わせを記述しておく。このファイルに基づいて、トラップのフィルタおよびプロセスの転送が行われる。デフォルトのファイルは、 <code>/etc/snmp/conf/enterprises.oid</code>
-m GROUP SPLIT	SNMP 要求の転送モードを指定する。デフォルトのモードは GROUP。2 つのモードの詳細は、30ページの「要求の送信」を参照
-p <i>port_number</i>	ポート番号を指定する。デフォルトのポート番号は 161。-p 1234 のように指定する
-r <i>filename</i>	マスターエージェントによって使用されるリソースファイル名のフルパスを指定する。リソースファイルには、マスターエージェントが呼び出したり、管理したりするサブエージェントについての情報を保存する。デフォルトのリソースファイルは、 <code>/etc/snmp/conf/snmpdx.rsrc</code> 。詳しくは、35ページの「エージェントリソース構成ファイル」を参照
-y	マスターエージェントのプロセスを呼び出して、復旧モジュールを呼び出す際の復旧インジケータシグナル。復旧プロセスでは、前のセッションのどのサブエージェントが実行状態のままかを検出する。実行状態にないサブエージェントは、マスターエージェントによって再度起動される

Enterprise Agents の構成

- 35ページの「Enterprise Agents の概要」
- 35ページの「エージェントリソース構成ファイル」
- 38ページの「エージェント登録ファイル」
- 41ページの「エージェントアクセス制御ファイル」
- 43ページの「マスターエージェント状態ファイル」

Enterprise Agents の概要

次のファイルは、構成用に使われます。

- マスターエージェントリソース構成ファイル
- エージェント登録ファイル
- エージェントアクセス制御ファイル
- マスターエージェント状態ファイル

エージェントリソース構成ファイル

エージェントリソース構成ファイルは、マスターエージェントによってのみ使われます。マスターエージェントが実行状態になった直後に、このファイルが読み込

まれます。このファイルには、マスターエージェントが管理するすべてのエージェントについての情報が保存されます。この構成ファイル内の各エントリには、サブエージェントの起動方法も指定されています。サブエージェントに構成ファイルがなくても、そのサブエージェントが動作可能になると、動的にマスターエージェントに登録されます。サブエージェントの動的な呼び出しと登録の詳細は、31ページの「サブエージェントについて」を参照してください。

エージェントに対してマスターエージェントによる呼び出しと静的な登録が選択されていると、そのエージェントは独自のリソース構成ファイルを持つことができます。このファイルには、サブエージェントの呼び出しに関する情報に加えて、サブエージェントに関連付けられている登録ファイルについての情報が含まれています。エージェント登録ファイルについては、38ページの「エージェント登録ファイル」を参照してください。

リソース構成ファイルの記述例を次に示します。

```
<ResourceFile> : Resource | Environment Resource
<Resource> : ``resource`` ``='`` ``{' ResourceList ``}'``
<ResourceList> : /*empty*/ | ResourceList ResourceItem
<ResourceItem> : ``{' StringList ``}'``
<Environment> : ``environment`` ``='`` ``{' EnvironmentList ``}'``
<EnvironmentList> : /*empty*/ | EnvironmentList
EnvironmentListItem
<EnvironmentListItem> : EnvironmentToken ``='`` Number
<EnvironmentToken> : ``poll-interval`` | ``Max-agent-time-out``
<Number> : Integer
<StringList>: StringItem | StringList StringItem
<StringItem> : StringToken ``='`` QuotedString
<StringToken> : ``registration_file`` | ``policy`` | ``command``
| ``type`` | ``user``
<QuotedString> : ``''`` AlphanumericString ``''``
```

snmpdx.rsrc ファイルと mibiisa.rsrc ファイルの例を次に示します。

構成ファイルで使う変数は、例のように記述します。コメント行は、# の文字で始めます。

```
snmpdx.rsrc
environment =
{
poll-interval = 5 # This is in seconds
max-agent-time_out = 1000000 # This is microseconds
}

mibiisa.rsrc
resource =
{
registration_file = /etc/snmp/conf/mibiisa.reg
security = ``/etc/snmp/conf/snmpd.conf``
type = ``legacy``
policy = ``spawn``
```

```
command = ``/usr/lib/bin/mibiisa -p $PORT``  
}  
}
```

environment グループ

environment グループは、マスターエージェントのふるまいを制御します。このグループには、次の2つの変数が含まれます。

- `poll-interval` - このフィールドの値は、秒単位で指定します。マスターエージェントは、指定した間隔をおいて SNMP メッセージの送受信以外の処理を実行します。また、すべてのエージェントが応答するかどうかを調べることによって、リソースファイルに変更が発生したことがわかると、機能を維持するために必要となる定型的な処理を行います。
- `max-agent-time-out` - このフィールドの値は、マイクロ秒単位で指定します。この変数は、サブエージェントが登録の際に要求できる最大許容タイムアウト値を表します。たとえば、マスターエージェントは、サブエージェントに要求を送るとタイムアウトの間は応答を受信するのを待ちます。このタイムアウト値は、登録ファイル内で指定します。または、動的登録によって指定されることもあります。エージェントがこのタイムアウト値にあまりにも大きな値を設定すると、マスターエージェントや他のエージェントで問題が生じます。そのような問題を避けるために、マスターエージェントがサブエージェントからの応答を待つタイムアウトの最大値を、マスターエージェントで指定する必要があります。そのタイムアウトの最大値が、この変数を使って指定できる最大許容タイムアウト値です。

resource グループ

resource グループの変数は、サブエージェントにだけ関係します。前述の例の構成ファイルには、2つのエントリが含まれています。各エントリはそれぞれのサブエージェントに対応しています。エントリには、値の指定された次の変数を記述できます。

- `type` - このフィールドには、`legacy` および `dynamic` の2つの値を指定できます。
- `registration_file` - このフィールドには、それぞれのサブエージェントの登録構成ファイルを指定します。マスターエージェントは、このフィールドに指定されたファイルからさまざまなエントリを読み込み、MIB テーブルに適切なエントリを作成します。このファイルの詳細は、38ページの「エージェント登録ファイル」を参照してください。このエントリは、すべてのレガシタイプ

のエージェントに必須です。この変数にフルパスで値が指定されていなければ、プログラムは、デフォルトのディレクトリ /etc/snmp/conf を調べます。

- **policy** - このフィールドには、load および spawn の 2 つの値を指定できます。値に load が指定されていると、マスターエージェントは登録エントリを読み込んで、MIB テーブルの行エントリを作成します。値に spawn が指定されていると、マスターエージェントは、command フィールドの指定内容に従ってそれぞれのサブエージェントを呼び出します。
- **command** - このフィールドには、サブエージェントの実行形式の名前を指定します。command にはフルパスを指定します。この変数にフルパスで値が指定されていなければ、プログラムは、デフォルトのディレクトリ /usr/lib/bin を調べます。command では、\$PORT マクロを使用して、サブエージェントが SNMP 要求を受信するポート番号を指定することができます。\$PORT の値は、各サブエージェントの登録ファイルの内容に従ってマスターエージェントが割り当てます。\$PORT マクロが必要とされるのは、レガシーエージェントやサブエージェントが、-p、-n、-port などのポートオプションに対して異なる引数を取ることがあるためです。
- **user** - サブエージェントは、このエントリに指定されたユーザーによって実行されます。

エージェント登録ファイル

エージェントにはそれぞれ、固有のエージェント登録ファイルがあります。このため、マスターエージェントとサブエージェントがそれぞれに変更を加えたファイルを持つこととなります。登録ファイルには、それぞれのエージェントに直接関係のある情報が収められています。また、エージェントの名前、エージェントによって管理されているサブツリー OID、要求のタイムアウト、選択されたポート番号も含まれています。このファイル内の各エントリの書式の例を次に示します。

```
<Config> : <Macro> <Agents>
<Macro> : ``macros`` ``='' ``{`` <MacrosList>``}''
<MacrosList> : <MacrosList> <MacroItem> | empty
<MacroItem> : label ``='' <SubidList>
<SubidList> : <SubidList> ``.'' <Subid> | <Subid>
<Subid> : ``mib2`` | ``sun`` | ``enterprise`` | identifier | number
<Agents> : ``agents`` ``='' ``{`` <AgentList> ``}''
<AgentList> : <AgentList> <AgentItem> | <AgentItem>
<AgentItem> : ``{`` <Name> <SubtreesTables> <TimeOut> <WatchDogTimer> <Port>``}''
<Name> : label ``='' quotestring
<SubtreesTables> : <SubtreesTables> | <Subtrees> | <Tables>
<Subtrees> : ``subtrees`` ``='' ``{`` <SubtreesList> ``}''
<SubtreesList> : <SubtreesList> ``','`` <SubtreeItem> | <SubtreeItem> | empty
```

```

<SubtreeItem> : <SubidsList>
<Tables> : ``tables'' ``=''' ``{'' <TableList>``}''
<TableList> : <TableList> <TableItem> | empty|
<TableItem> : ``{'' <Table> <Columns> <Indexs> ``}''
<Table> : ``table'' ``=''' <SubidsList>
<Columns> : ``column'' ``=''' <Range>
<Range> : ``['' number ``]'' | number
<Index>s : ``indexs'' ``=''' <Range>
<TimeOut> : ``timeout'' ``=''' number
<WatchDogTimer>: ``watch-dog-time'' ``=''' number
<Port> : ``port'' = number

```

登録ファイルの名前には、拡張子を付けることができます。その場合、reg という拡張子にすることをお勧めします。実際のサブツリーのファイルの例を次に示します。

```

macros = {
applicationTable = mib-2.27
sun = enterprise.42
}
agents = {
{
name = ``ExampleAgent''
subtrees = { mib-2, sun }
tables = {
{ #begin table
table = applicationTable
columns = [ 2 -15 ]
indexes = [ 2 -3 ]
} #end table
} #end of tables
timeout = 20000 # Optional. Each SNMP request time out. This is
in microseconds.
watch_dog_time = 300 # This is in seconds
port = 4000 # Optional
}
} #end of agents

```

この構成ファイルは、次のような 2 つの情報のグループで構成されています。

- **macros** – エージェントグループで使うマクロが含まれます。
- **agents** – 多くの変数と、定義内でさらに「テーブル化された」グループが含まれます。

agents グループで使われる変数は、次のとおりです。

- **name** - この変数には、サブエージェントの名前を指定します。別のプロセスとして呼び出される複数のエージェントの実行形式ファイルが同じでも、エージェント名は一意でなければなりません。マスターエージェントは、エージェントテーブル MIB 内のキーとしてエージェント名を使います。
- **subtrees** - この変数には、特定のエージェントによって管理されているサブツリー OID のリストを指定します。サブエージェントは複数のツリーを管理でき

ます。前述のサブツリーの例では、ExampleAgent という名前のエージェントが、mib-2 と sun というサブツリーを管理しています。

- **tables** - MIB テーブル全体またはテーブルの一部を管理するように、サブエージェント登録ファイルを構成できます。tables グループには、テーブル名、カラム番号、および特定の行番号 (indexes) を指定します。前述のサブツリーの例では、ExampleAgent が、サブツリー mib-2 と sun のほかに、そのアプリケーションテーブルのカラム 2~15 と行 2~3 を管理しています。
- **timeout** - timeout 変数はマスターエージェントに登録されます。マスターエージェントは、timeout で指定された時間 (単位: マイクロ秒)、SNMP 要求に対する応答を受信するのを待ちます。timeout には、マスターエージェントのリソース構成ファイルであらかじめ定義されている max_agent_time_out を超えない適切な値を設定します。
- **watch_dog_time** - マスターエージェントはこのタイムアウトを使って、サブエージェントが実行状態であるかどうかを判断します。watch_dog_time の間にマスターエージェントとサブエージェントのあいだにアクティビティがない場合にだけ、マスターエージェントはサブエージェントに対してポーリングを行います。
- **port** - マスターエージェントから SNMP 要求を受信するためにサブエージェントが待機しているポート番号を指定します。この変数は省略可能です。通常、サブエージェントは、この変数に値を割り当てません。この変数が構成ファイルにない場合、マスターエージェントは、オープンしていないポートを検索し、そのポート番号を使用して各サブエージェントを呼び出します。port に値が割り当てられている場合、マスターエージェントは、その特定のポート番号を使用してサブエージェントを呼び出します。サブエージェントは、マスターエージェントのリソースファイル内の command 変数に従って呼び出されます。

注 - Solstice Enterprise Agents のライブラリを使って開発された動的なエージェントは、マスターエージェントで呼び出す必要はありません。この場合は、マスターエージェントのリソースファイル内には、そのようなエージェントに対するエントリがないことがあります。このようなエージェントは、エージェント自身で利用可能なポートをオープンします。

エージェントアクセス制御ファイル

エージェントアクセス制御ファイルは、SNMP 関連のコミュニティ情報を保存する構成ファイルです。各サブエージェントとマスターエージェントには、独自のアクセス制御ファイルがあります。このファイル名には拡張子を付けることができます。拡張子は `acl` にすることをお勧めします。このファイルは、`/etc/snmp/conf` というディレクトリに保存する必要があります。

アクセス制御構成ファイルの構文の例を示します。

```
<snmp_security> : <acls> <trap_block>
<acls> : /*empty*/ | ``acl'' ``=''' {<acls_list> }
<acls_list> : /*empty*/ | <acls_list> <acl_item>
<acl_item> : {<communities_stmt> <acl_access> <hosts> }
<communities_stmt> : ``communities'' ``=''' <communities_set>
<communities_set> : <communities_set> , <community_elem> |
<community_elem>
<community_elem>: alphanumeric_string
<acl_access> : ``access'' ``=''' <acl_access_type>
<acl_access_type> : read-only | read-write
<hosts> : ``managers'' ``=''' <hosts_list>
<hosts_list> : <hosts_list> , <host_item> | <host_item>
<host_item> : alphanumeric_string
<trap_block> : ``trap'' ``=''' { <traps_list> }
<traps_list> : /*empty*/ | <trap_list> < trap_item>
<trap_item> : { <trap_community_string> <trap_interest_hosts>
<enterprise_list> }
<trap_community_string> : ``trap-community'' ``=''' alphanumeric_string
<trap_interest_hosts_list> : <trap_interest_hosts_list> ,
<trap_interest_host_item> |
<trap_interest_host_item>
<trap_interest_host_item> : alphanumeric_string
<enterprise_list> : /*empty*/ | <enterprise_list> <enterprise_item>
<enterprise_item> : { <enterprise_stmt> <trap_number_stmt> }
<enterprise_stmt> : ``enterprise'' ``=''' quoted_alphanumeric_string
<trap_number_stmt> : ``trap-num'' ``=''' <trap_number_list>
<trap_number_list> : <trap_number_item>
<trap_number_item> : <trap_range>
<trap_range> : integer - integer | integer
```

アクセス制御リストファイルの例を示します。

```
acl = {
{
communities = public, private
access = read-only
managers = hubble, snowbell, nanak
}
{
communities = jerry
access = read-write
managers = hubble, telescope
}
```

```

}
trap = {
  {
    trap-community = SNMP-trap
    hosts = hubble, snowbell
    { enterprise = ``Sun``
      trap-num = 1, 2-5
    }
  }
  {
    enterprise = ``3Com``
    trap-num = 4 }
  }
  {
    trap-community = competitor-trap
    hosts = hp_server, ibm_server, sgi
    {
      enterprise = ``sun``
      trap-num = 1,3 }
    {
      enterprise = ``snmp``
      trap-num = 1-32
    }
  }
}
}

```

アクセス制御リストファイルには、2つのグループの構成変数が保存されています。

- **acl** - このグループの変数は、コミュニティ名、アクセス権、および SNMP 要求を受け付けるホスト名の3つが1組となって定義され、それが多数記述されます。なお、あらかじめ設定しておいたコミュニティ名が含まれていなければ、この SNMP 要求は受け付けられません。前述のアクセス制御リストファイルの例では、**public** と **private** というコミュニティ名を含むホスト **hubble**、**snowbell**、および **nanak** からの GET および GET_NEXT SNMP 要求だけが受け付けられます。このグループには、3つの組み合わせを複数指定できます。

マスターエージェントには、SNMP PDU を受け取るための適切なアクセス権とコミュニティがあります。ただし、同じ SNMP PDU がサブエージェントに転送される場合、マスターエージェントは受信するための適正な権限がなかったり、適正なコミュニティが含まれていない場合、その PDU を拒否することがあります。マスターエージェントに適切なコミュニティとアクセス権がなければ、サブエージェントに SNMP PDU を受信するためのアクセス権とコミュニティがあったとしても、そのような PDU はサブエージェントに到達することはありません。

- **trap** - この変数のグループは、サブエージェントから受信するトラップを送信したり、転送したりするための情報で構成されます。マスターエージェントは、トラップを転送する際にこのグループで指定された情報を使います。この情報に

は、構成されたトラップ番号を送るためのホスト名を指定します。トラップの PDU には、指定したトラップのコミュニティが含まれます。これらのトラップは、最初にサブエージェントによって生成され、次にマスターエージェントに送信されます。

マスターエージェント状態ファイル

マスターエージェントの状態ファイルには、マスターエージェントによって生成されたさまざまなサブエージェントに関する情報が保存されています。マスターエージェントのみがこのファイルを使用します。マスターエージェントはこのファイルに情報を動的に追加します。そのため、その情報を自分で編集する必要はありません。マスターエージェントがサブエージェントのプロセスを生成するたびに、このファイルにエントリが作成されます。このファイルは、マスターエージェントに障害が生じて動作しなくなってしまった場合に、マスターエージェントを復旧するためのものです。マスターエージェントを再起動すると、このファイル内のエントリによって、以前に作成されたサブエージェントや対応するポート番号が示されます。マスターエージェントは、このファイル内の各エントリを読み込み、それをマスターエージェントのリソース構成ファイル内のエントリと比較します。そのエントリがリソースファイル内になければ、マスターエージェントはそのプロセスを終了します。エントリが両方のファイル内に存在する場合、マスターエージェントはポートを使用してサブエージェントへのアクセスを試みます。

MIB の発行

Solstice Enterprise Agents 技術を有効利用するために、サン・マイクロシステムズ社固有のエンタープライズ MIB OID 配下にテーブルが 3 つ定義されています。次のテーブルの例は、すべてのサブエージェントの管理特性を提供するためのものです。このテーブル内の情報で、サブエージェントに固有の情報を提供しています。このテーブルには、サブエージェントの名前とサブエージェントのポート番号が収められています。各サブエージェントによって管理されているサブツリーの OID の例は示してありません。

MIB の例

すべての種類の MIB 変数を含む MIB の例を次に示します。MIB にはテーブルも組み込まれています。この MIB が `mibcodegen` を通じて実行されると、適切な MIB データベースとスタブコードが生成されます。これによって、この MIB のサブエージェントが構築されます。

```
DEMO-MIB DEFINITIONS ::= BEGIN

    IMPORTS
        OBJECT-TYPE, Counter32, Gauge32
        FROM SNMPv2-SMI
    DisplayString, TimeStamp
        FROM SNMPv2-TC;

    mib-2          OBJECT IDENTIFIER ::= { mgmt 1 }
    sun  OBJECT IDENTIFIER ::= { enterprises 42 }
    demo OBJECT IDENTIFIER ::= { sun 1000 }

    --
    -- Some objects
    --
    demoString OBJECT-TYPE
        SYNTAX DisplayString
        MAX-ACCESS read-write
        STATUS current
        DESCRIPTION
            "A read-write object of type String."
        ::= {demo 1}

    demoInteger OBJECT-TYPE
        ::= {demoTable 1}
    DemoEntry ::= SEQUENCE {
        demoEntryIndex
            INTEGER,
        demoEntryString
            DisplayString,
        demoEntryInteger
            INTEGER,
        demoEntryOid
            OBJECT IDENTIFIER }
    SYNTAX INTEGER {
        up(1),
        down(2) }
        MAX-ACCESS read-write
        STATUS current
        DESCRIPTION
            "A read-write object of type Integer."
        ::= {demo 2}

    demoOid OBJECT-TYPE
        SYNTAX OBJECT IDENTIFIER
        MAX-ACCESS read-write
        STATUS current
        DESCRIPTION
            "A read-write object of type Oid."
        ::= {demo 3}
    -- A table composed of some columns
```

```

demoTable OBJECT-TYPE
    SYNTAX SEQUENCE OF DemoEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "A table."
    ::= {demo 10}

demoEntry OBJECT-TYPE
    SYNTAX DemoEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "An entry in the table demoTable."
    INDEX {demoEntryIndex}
demoEntryIndex OBJECT-TYPE
    SYNTAX INTEGER (1..2147483647)
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "An index to uniquely identify the entry."
    ::= {demoEntry 1}

demoEntryString OBJECT-TYPE
    SYNTAX DisplayString
    MAX-ACCESS read-write
    STATUS current
    DESCRIPTION
        "A read-write column of type String."
    ::= {demoEntry 2}

demoEntryInteger OBJECT-TYPE
    SYNTAX INTEGER {
        up(1),
        down(2) }

    MAX-ACCESS read-write
    STATUS current
    DESCRIPTION
        "A read-write column of type Integer."
    ::= {demoEntry 3}

demoEntryOid OBJECT-TYPE
    SYNTAX OBJECT IDENTIFIER
    MAX-ACCESS read-write
    STATUS current
    DESCRIPTION
        "A read-write column of type Oid."
    ::= {demoEntry 4}
demoTrap TRAP-TYPE
    ENTERPRISE sun
    VARIABLES { demoInteger, demoString, demoOid}
    DESCRIPTION
        " Trap for testing."
    ::= 2
demoColdLinkTrap TRAP-TYPE
    ENTERPRISE snmp
    DESCRIPTION
        " Trap for testing."

```

```
 ::= 0  
END
```

DMI の使用方法

- 47ページの「DMI の使用方法の概要」
- 47ページの「DMTF とは」
- 48ページの「DMI の機能」
- 49ページの「DMI のアーキテクチャ」
- 53ページの「DMI API ライブラリ」
- 53ページの「MIF から MIB への変換コンパイラ」
- 54ページの「マッパー」

DMI の使用方法の概要

この章では、デスクトップ管理タスクフォース (DMTF)、そのグループの運営管理、およびデスクトップ管理インタフェース (DMI) について説明します。

DMTF とは

DMTF は、8 社の共同成果として 1992 年 5 月に設立されました。その 8 社とは、Digital Equipment Corporation™、Hewlett-Packard™、IBM™、Intel™、Microsoft™、Novell™、Sun™、および SynOptics™ で

す。DMTF の目的は、デスクトップの管理のための基本的な解決法を提供することです。

DMTF は、デスクトップ PC やサーバー上の管理可能な製品と管理アプリケーションとの間での通信を処理するための標準インタフェースを開発しました。

この標準インタフェースは、DMI と呼ばれます。DMI の詳細は、<http://www.dmtf.org> の『*DMI Specification Version 2.0*』を参照してください。

DMI の特徴は次のとおりです。

- 特定のオペレーティングシステム、ハードウェアプラットフォーム、または管理プロトコルに依存しない
- ベンダーが容易に採用できる
- 非常に簡単なものから非常に複雑で拡張可能なものまで、幅広い範囲の製品に適応できる
- 既存の管理プロトコルとリモートプロトコルにマップできる

DMI の機能

SEA の DMI の機能には、次のようなものがあります。

- コンポーネントインストールメンテーションや管理アプリケーションの動的なインストールや削除
- MIF データに対する全実行時アクセスの管理
- 各 MIF ファイル内に、少なくとも 1 つのグループ (コンポーネント ID グループ) が必ず存在することを保証する
- 必要に応じた、コンポーネントインストールメンテーションの起動の管理
- コマンドの分割。管理アプリケーションが、1 つのコマンドで構成要素に対して 1 つ以上の属性値を要求した場合、SP は属性ごとにコンポーネントインストールメンテーションにコマンドを送る
- コマンドをコンポーネントインストールメンテーションに順番に渡し、確実に完了するまで実行されるようにする。特定のコンポーネントインストールメンテーションに対する複数の要求を待ち行列に入れる
- イベントとインジケーションの登録機能とフィルタ機能の提供

- 登録機能とフィルタ機能に基づいて、インジケーションを管理アプリケーションに対して転送したり、そのインジケーションの転送前に、受け取ったインジケーションにタイムスタンプ情報を付加したりする
- MIF データベースに対する構成要素のインストールや削除時に、インジケーションを受け取るように登録されているすべての管理アプリケーションへインジケーションを送信
- 管理アプリケーションからは、ID 1 の構成要素として認識される。構成要素として、標準の ComponentID グループをサポートしている必要がある。さらに、DMI SP は、登録機能と標準のグループのフィルタ機能をサポートしている必要がある。また、構成要素と同様に、ComponentID グループ以外のほかのグループを定義できる

DMI のアーキテクチャ

SEA 製品に含まれている DMI に基づいた解決法は、いくつかの方法で使用できます。たとえば、DMI を SNMP サブエージェントとして扱うこともできます。さらに、SP と直接やりとりするように、DMI ベースの管理アプリケーションを記述することもできます。

SNMP サブエージェントモードでは、SNMP 要求は DMI 要求にマップされ、DMI SP と通信が行われます。ダイレクトモードでは、管理アプリケーションは、DMI を使って SP と直接通信できます。

図 5-1 に、DMI と Enterprise Agents との関係を表すアーキテクチャ全体を示します。

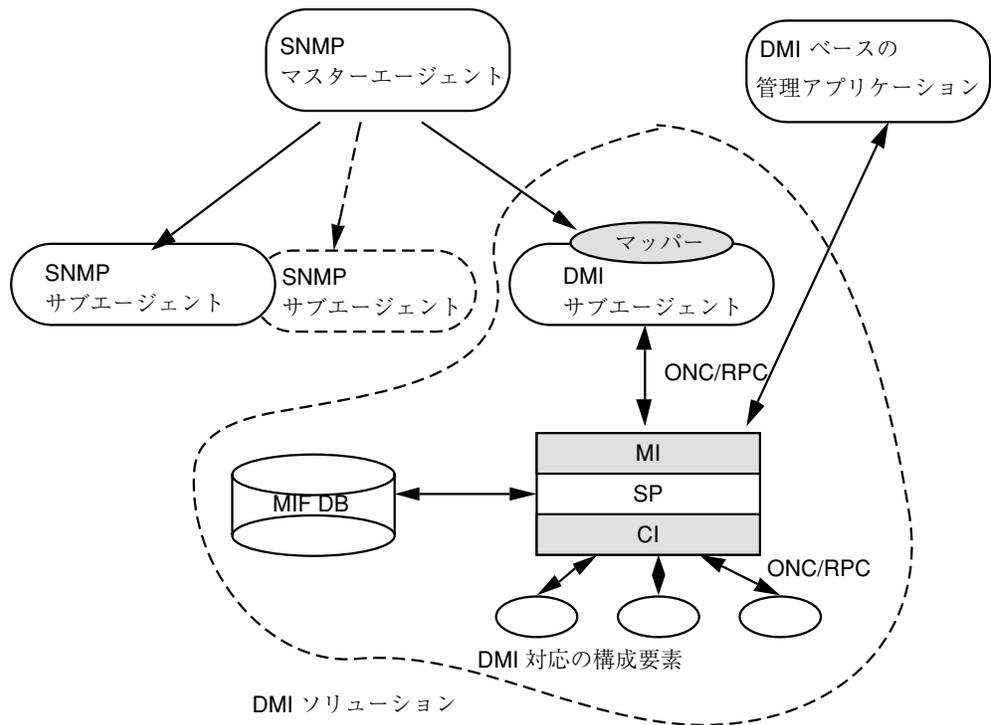


図 5-1 DMI と Enterprise Agents

DMI サービスプロバイダ

DMI SP は、DMI の核となるものです。管理アプリケーションとコンポーネントインストールメンテーションは SP を通じて通信を行います。SP は、管理アプリケーションから指定された構成要素に対する要求を調整したり、仲介したりします。SP は、コンポーネントインタフェース (CI) と管理インタフェース (MI) のランタイム管理を行います。ランタイム管理には、構成要素のインストール、MI レベルと CI レベルの登録、要求の順序管理と同期、CI のイベント処理、一般的なフロー管理やこれらの管理に必要なすべての動作などがあります。

図 5-2 に、1 つのシステム内に存在する要素、または直接接続されている要素を示します。管理アプリケーションは DMI のブラウザとしても使えます。



ハードウェアおよびソフトウェアの構成要素

図 5-2 DMI サービスプロバイダ

DMI SP は、次の 4 つのモジュールで構成されています。

- SP プロセス
- MI (DMI SP と通信するために管理アプリケーションが使うインターフェース)
- コンポーネントインターフェース (コンポーネントインストールメンテーションのコードを使用するための SP のインターフェース)
- SP に関連付けられている MIF データベース

管理インターフェース

SEA の MI の機能には、次のようなものがあります。

- 管理アプリケーションは、MI を介して DMI と通信を行います。管理アプリケーションは、DMI の Get、Set、および List コマンドを実行することによって、システム内の構成要素に関する情報を要求します。また、イベント通知と適正なフィルタ機能を SP に登録します。
- イベントが生成されると、DMI SP は、その登録テーブルとフィルタテーブルを調べます。フィルタプロセスを通過したイベントは、DMI SP によって転送されます。イベントは、これらのイベントを受信できるように SP に登録されているすべての管理アプリケーションに転送されます。

コンポーネントインターフェース

構成要素は、CI を通じて DMI と通信を行います。構成要素のサブエージェントは、装置やアプリケーションなどのそれぞれの構成要素を管理するためにエンドユーザーによって作成されます。CI によって提供される機能には、次のようなものがあります。

- 登録 – 構成要素を DMI SP に登録します。
- イベントの送信 – コンポーネントインストールメンテーションは、SP にインジケーションブロックが送信され、処理されます。DMI マッパーは、DMI インジケーションを SNMP トラップに変換します。
- 構成要素は、DMI SP 要求に応じて、そのサブエージェントによってさまざまな属性値に対して Get や Set 処理を行います。

MIF データベース

SEA の MIF データベースの機能には、次のようなものがあります。

- SP には、1 つの MIF データベースが関連付けられています。それぞれの構成要素には、管理可能な特性を記述するための MIF ファイルがあります。構成要素がシステムに最初にインストールされる時、MIF が MIF データベースに登録されます。SP は、MIF データベースへのすべてのアクセスを制御します。
- MIF データベースに対する MIF の修正機構は提供されていません。MIF に変更が必要な場合は、インストールを解除してから標準のテキストエディタツールを使って変更してインストールし直す必要があります。
- MIF データベースから MIF のインストールや削除を行う場合、SP によって、すべての登録された管理アプリケーションにインジケーションが発行されます。

DMI サービスプロバイダの起動

DMI SP をインストールすると、起動時にスクリプトファイルが DMI SP を呼び出します。DMI SP は、次のようなオプションを使って起動されます。

```
dmisspd [-h] [-c config_dir] [-d debug_level]
```

表 5-1 IDMI SP 呼び出しの引数

引数	定義
-h	コマンド行の使用方法を表示する
-c <i>confid_dir</i>	dmispd.conf 構成ファイルを含むディレクトリのフルパス
-d <i>debug_level</i>	デバッグモードでは、プロセスはデーモンとして動作せずに、表示画面にトレースメッセージを表示する。 <i>debug_level</i> (1~5) に応じて、特定の情報を印刷する

DMI API ライブラリ

SEA パッケージに含まれている DMI API ライブラリは、DMI を使って管理アプリケーションを開発するためのプロシージャを含む C のライブラリです。このライブラリは、ユーザーがサブエージェントを作成するためのコンポーネントインタフェースを提供します。これには、構成要素を管理するためのコンポーネントインストールメンテーションも含まれます。さらに、DMI API によって、MIF データベースへの構成要素のインストールと SP のコンポーネントインタフェースの呼び出しプロセスが単純になります。

MIF から MIB への変換コンパイラ

このユーティリティは、DMI MIF ファイルを SNMP フォーマットに変換し、マップ構成ファイルを生成します。SunNet Manager、Enterprise Manager などのネットワーク管理アプリケーションは、MIB を使って DMI ベースの構成要素 MIF を管理できます。マッププロセスを実行するときは、マップファイルを使います。マップファイルは、SNMP ベースの MIB 変数を DMI ベースの MIF 属性にマップするときに使われます。

```
miftomib ''[mifname=] [value value ...]'' mifpathname [mibpathname]
```

表 5-2 MIF から MIB への変換コンパイラの引数

引数	定義
<i>mifname</i>	生成される MIB オブジェクトの名前
<i>value</i>	スペースで区切られた 1 つ以上の整数
<i>mifpathname</i>	MIF ファイル
<i>mibpathname</i>	生成された MIB ファイル

マッパー

マッパーは、DMI 管理アプリケーションとして動作する SNMP サブエージェントです。マッパーは、管理インターフェースを使って DMI SP に管理要求を送ります。また、SP を通じて構成要素からのイベントを処理し、それらをマスターエージェントに渡します。そのため、DMI の利用可能な構成要素は、ほかの SNMP で管理されている構成要素と同じように見えます。図 5-3 に、マッパーと構成要素の通信方法を示します。

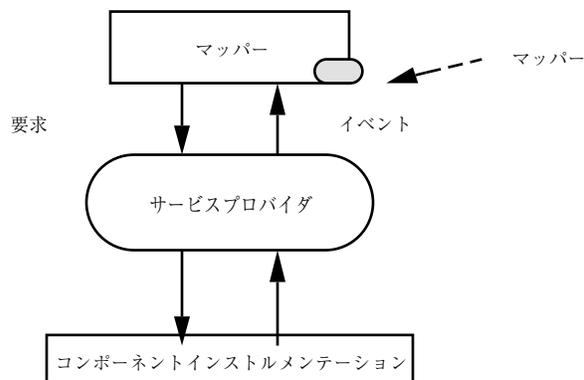


図 5-3 DMI マッパーと DMI 構成要素

次の節では、動作中のサブエージェントの作業について説明します。

サブエージェントの初期化と再インストール

- サブエージェント自身を SP に登録します。これによって、MIF データベースへのアクセスや、構成要素からのインジケーションなどのサービスが SP によって提供されます。
- マップ用の変換テーブルを作成します。変換テーブルは、各構成要素に対して作成されたマップファイルを読み込むことによって作成されます。マップパーが最初に起動されたときに変換テーブルが構築されます。そのあとで、新しい構成要素がインストールされ SP に登録されると、DMI SP はそのマップパーにイベントを送信します。次にマップパーが、新しいマップファイルのエントリをマップテーブルに追加します。
- マスターエージェントとサブエージェントとの接続を確立します。また、サブエージェント自身をマスターエージェントに登録します。
- SNMP と DMI とのマップと、SNMP OID の DMI オブジェクトへの動的な変換に使われる変換テーブルを構築します。

マップパーの呼び出し

マップパーは、次のようなオプションによって呼び出されます。

```
snmpXdmid -s hostname [-h] [-c config_dir] [-d debug_level]
```

表 5-3 DMI SP 呼び出しの引数

引数	定義
-h	コマンド行の使用方法を表示する
-s <i>hostname</i>	SP が動作しているホストの名前。デフォルトではそのローカルホスト名
-c <i>config_dir</i>	構成ファイル <code>snmpXdmid.conf</code> を含むディレクトリのフルパス
-d <i>debug_level</i>	デバッグモードでは、プロセスはデーモンとして動作せずに、表示画面にトレースメッセージを表示する。 <i>debug_level</i> (1~5) に応じて、特定の情報を印刷する

DMI による SNMP の使用

- 57ページの「DMI による SNMP の使用の概要」
- 59ページの「SNMP と DMI との通信」
- 59ページの「SNMP マスターエージェントへの構成要素の登録」
- 60ページの「Solaris での DMI マッパーの実行」
- 61ページの「DMI マッパーの機能」
- 63ページの「SNMP 要求の DMI への変換」
- 71ページの「DMI マッパー構成ファイル」
- 72ページの「MIB ファイルの生成」

DMI による SNMP の使用の概要

Solstice Enterprise Agents (SEA) の技術を採用すれば、SNMP で通信する管理アプリケーションから、snmpXdmiid という DMI マッパーを介して DMI の利用可能な構成要素に対してアクセスできます。SNMP は、プロトコルデータユニットを使って、ネットワークに分散しているエージェントと管理アプリケーションとの間で情報の通信を行います。標準の MIB には、SNMP 管理アプリケーションによってすべての被管理オブジェクトが記述されます。エージェントプログラムは、管理アプリケーションの要求に従って MIB オブジェクトの値を提供したり、変更したりします。

DMI マッパー snmpXdmid は、SNMP 管理アプリケーションと通信するために、DMI 情報を SNMP MIB フォーマットに動的に変換するマップ機能を提供します。SNMP 管理アプリケーションは、snmpXdmid に要求を送信します。すると、その SNMP 要求は DMI 要求に順番に変換されます。DMI 応答が返送されると、今度は逆変換が行われます。この技術を採用することによって、SNMP 管理アプリケーションは、DMI の利用可能な構成要素の管理に積極的に関与することができます。

この章では、DMI と SNMP の結び付きを示し、SNMP MIB と DMI MIF の間でデータをマップする方法を説明します。

図 6-1 に、どのようにして snmpXdmid が Solstice Enterprise Agents のほかの部分と協同して動作するのかを示します。

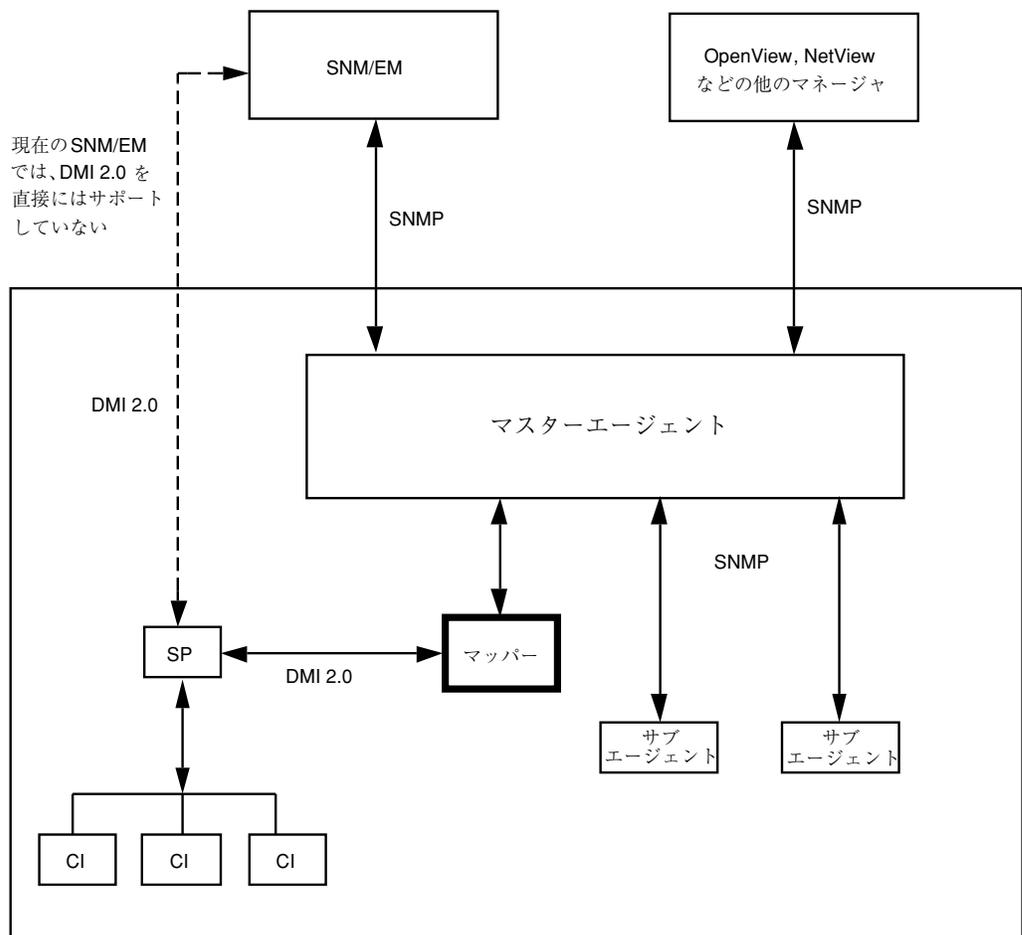


図 6-1 snmpXdmid の概要

SNMP と DMI との通信

SNMP 管理アプリケーションと DMI の間の通信経路は、次の節で説明するように、SNMP マスターエージェントと DMI マッパーで構成されます。

SNMP マスターエージェント

SNMP マスターエージェントは、システム内の管理アプリケーションと登録されたサブエージェント間の SNMP 要求とその応答を処理します。Solstice Enterprise Agents によってインストールされた `snmpd` 実行形式ファイルは、システムにすでにインストールされている `snmpd` ファイルと置き換えられます。

SNMP マスターエージェントは、SNMP PDU を介してシステム内のサブエージェントと通信を行います。サブエージェントとしての役割を果たすプロセスでは、MIB サブツリーを SNMP エージェントに登録できます。登録された変数に対する要求をマスターエージェントが受信すると、その要求はすべて `snmpdXdmid` を経由してサブエージェントに渡されます。次にその要求が実行され、マスターエージェントに応答が返送されます。

DMI マッパー

DMI マッパーは、システム内の SNMP マスターエージェントからの要求と SNMP マスターエージェントへの応答を処理します。マッパーは、特定の MIB 変数に対する SNMP Get 要求などの要求を受信すると、その MIB 変数を対応する DMI MIF 属性に変換します。システム内の DMI 構成要素より送信されたインジケーションは、SNMP トラップに変換され、管理アプリケーションに送信されます。MIB 変数の MIF 属性への変換やインジケーションのトラップへの変換は、マッパーが SNMP マスターエージェントに登録した構成要素にかぎり実行されます。

SNMP マスターエージェントへの構成要素の登録

`snmpXdmid` によって構成要素のマッピングが行われるようにするには、構成要素の MIF ファイルと、SNMP オブジェクト識別子 (OID) の接頭辞とが相互に関係付けら

れている必要があります。OID の接頭辞は、DMI 構成要素を SNMP マスターエージェントに結び付けるための登録点です。

OID と構成要素との関係付けは、構成要素のエントリをマップファイルに追加することによって行われます。.MAP ファイルの書式については、67ページの「MIF から MIB へのマップ」を参照してください。Solaris 環境では、.MAP ファイルは /var/dmi/map ディレクトリにあります。

注 - DMI マッパーがマップ処理を適切に実行するために、マップファイルの構成要素名のフィールドのエントリが、MIF ファイルのその構成要素に対して指定された名前の値と一致していなければなりません。このとき、文字列のスペースや大文字小文字なども一致している必要があります。

マップファイルに変更が行われる場合は、DMI マッパーが現在のマップ情報を管理できるようにするために、次のどちらかの処置を行う必要があります。

- .MAP ファイルが変更されたあとに、新しい構成要素が登録される場合、snmpXdmid は自動的に .MAP ファイルを読み込み直すため、明示的な処置は必要ありません。
- 構成要素は登録されずに、.MAP ファイルだけが変更される場合は、snmpXdmid を停止して、起動し直す必要があります。

注 - マップファイル内で異なるエンタープライズ ID を持つ同じ構成要素名が繰り返される場合は、DMI マッパーはその構成要素を 2 回マップします。

Solaris での DMI マッパーの実行

Solaris 環境で DMI マッパー snmpXdmid を実行するときは、次の操作を行なって、SNMP 情報の流れが必ず適正になるようにします。

- マッパーのトラップを送信するそれぞれのホストに対するエントリを snmpdx.ac1 ファイルに追加します。
- 管理アプリケーションがマッパーへの要求に使用するコミュニティ名に、必ず書き込み可能なアクセス権を与えます。これは、snmpdx.ac1 ファイルで指定します。コミュニティ名に書き込み可能なアクセス権が与えられていないと、そのコミュニティを使う管理アプリケーションが実行する Get および Set の操作は失敗します。

- `snmpdx.ac1` では、0~16 のトラップ ID が Sun エンタープライズの下で監視されるようにします。`snmpXdmid` をシステムにインストールすると、これは自動的に処理されます。

`snmp.conf` ファイルには、エントリの書式に関する記述が含まれています。これは、エントリを追加するときの参考になります。`snmp.conf` ファイルは、`/etc` ディレクトリにあります。

DMI マッパーの機能

サブエージェントは、システム内の DMI 構成要素を SNMP マスターエージェントに登録します。マッパーが正しくマップ処理を行うには、SNMP エージェントからの要求を処理するときに、サブエージェントによって使用されるマップファイル内で、構成要素の MIF 構造が認識される必要があります。マップファイルには、構成要素に対する一意の SNMP OID が保存されています。この OID は、マスターエージェントの登録点として使われます。

マップファイルを生成するには、次の作業のどちらかを行います。

- テキストエディタを使ってファイルを作成する
- `miftoMib` ユーティリティを使って、マップファイルと、構成要素 MIF に対応する SNMP MIB ファイルを生成する

サブエージェントの処理プロセスには、次のような状態があります。

- 初期化
- 通常のコ作
- 例外の報告
- 終了

これらの状態によって、一般的な処理の流れが規定されます。マッパー `snmpXdmid` は、デーモンとして動作し、通常、SNMP 要求を待ち、要求を受信するとすぐにその要求を処理し、応答を返送し、次の SNMP 要求を待ちます。また、マッパーは DMI SP からインジケーションも受信し、デフォルトでは、これらのインジケーションを SNMP トラップとして転送します。

▼ サブエージェントの初期化と再初期化

通常、マップパーは、起動スクリプトを介してシステムのブート時に起動されます。マップパーの起動は、システムの起動段階のあとの方で行う必要があります。マップパーの起動によって、SNMP マスターエージェントと DMI (SP) が初期化されます。初期化の際には、snmpXdmid とマスターエージェントの間、および snmpXdmid と DMI SP の間の両方で動的に登録が行われます。

1. 管理アクティビティが開始される前に、マップパーは、管理インタフェース (MI) の登録の呼び出しを使用して、サービスプロバイダ (SP) にマップパー自身を登録します。

これによって、SP が、MIF データベースへのアクセスなどのサービスを提供できるようになります。また、マップパーは、インジケーションを受信できるように、SP の登録テーブルとフィルタテーブルにエントリを追加します。

2. マップパーが変換テーブルを構築します。

変換テーブルには、.MAP ファイル内にある、それぞれ一意の SNMP OID 接頭辞と構成要素の名前の組み合わせに対するマスターエージェントの登録点が保存されます。これらの変換テーブルを構築するときに、マップパーは /var/dmi/map ディレクトリにあるすべての .MAP ファイルを検索します。

これらの構成要素の識別子と、それぞれの構成要素に関連付けられているグループの識別子は、変換のために保存されます。インストール済みの構成要素でも、MAP ファイルに登録されていないためにプログラムで検索できないものには、SNMP エージェントはアクセスできません。構成要素のインストール、またはインストールの解除のたびに、変換テーブルは適切に調整されます。

新しい構成要素が SP に登録されると、マップパーはすべてのマップファイルを読み込み直して、変換テーブルが常に最新になるようにします。

3. マップパーがマスターエージェントに接続されます。

この接続で通信が行われる前に、マップパーは接続を確立し、マスターエージェントに登録する必要があります。

次の定義を使って登録が行われます。

- サブエージェント ID
- エージェント状態 = ACTIVE
- タイムアウトの値
- サブエージェント名

- サブエージェントのアドレス

4. この時点で、エージェントは、管理する **MIB** のオブジェクト **ID** を登録します。
.MAP ファイルから取得される MIB OID の接頭辞が登録に使われます。構成要素がインストールされていなくても、その構成要素名に関連付けられている OID の接頭辞を登録することもできます。

初期化の際には、DMI とそのインタフェースが機能できる状態で、しかも変換テーブルが正しければ、コールドスタートトラップがエージェントに送信されます。トラップの詳細は、65ページの「例外の報告」を参照してください。

再初期化は、重大なエラーまたは潜在的に重大なエラーが検出されたときに行います。この場合、DMI とマスターエージェントとの通信中にそのどちらかでエラーが発生した可能性があります。DMI、マスターエージェント、およびテーブルの初期化をやり直すと、ウォームスタートトラップがマスターエージェントに送信されます。

SNMP 要求の DMI への変換

SNMP マスターエージェントは要求を受信すると、そのオブジェクトが、マッパーを登録したサブツリー内にあるかどうかを判断します。そのオブジェクトがサブツリー内にある場合は、マッパーに SNMP 要求を送ります。マッパーはそのパケットを受け取ると、次の SNMP の種類に従ってパケットを解析します。

- GET
- GETNEXT
- SET

SNMP 要求の DMI への変換について、表 6-1 に示します。

表 6-1 SNMP 要求の DMI への変換

SNMP	オブジェクト	DMI	データ
{verb}	{noun}	{verb}	{noun}
get	OID	MI_cmds	ID と Key

表 6-1 SNMP 要求の DMI への変換 続く

SNMP	オブジェクト	DMI	データ
getnext			comp ID/group ID/attr ID
set/commit/undo			

構成要素、グループ、または属性のリストを問い合わせたり、個別の属性を取得 (Get) したり設定 (Set) したりするための MI コマンドを実行することによって、MIF データベースにアクセスできます。

マッパーはマスターエージェントから GET 要求を受け取ると、次の処理を行います。

1. 変換テーブル内のエントリの有無に応じて、その要求が MIF に対するものかまたは MIB に対するものかを判断します。
2. OID の残りの部分をパース処理することによって、妥当性をチェックします。
3. 指定されたグループと構成要素があるかどうか変換テーブルを検索します。
4. OID に DMI テーブルインデックスが指定されている場合は、それを DMI キー形式に変換します。
5. DMI からオブジェクト値を検索します。オブジェクト値が見つかると、サブエージェントはそのオブジェクト値をエージェントに渡します。

GETNEXT 要求の処理中に、サブエージェントは、字句解析上の順序が維持されていることを確かめる必要があります。SMI オブジェクトとして返される属性を検出するまでに、MIF データベースの検索規模がかなり大きくなる場合があります。その属性に固有なエラーなどの小さなエラーが原因で、目的の属性の値が利用できないときは、/RFC1448/ によって規定されている genErr が返されます。次のオブジェクトがサブエージェントのツリーになれば、その要求と同じオブジェクトのインスタンスとともに、値 noSuchName が返されます。

DMI 属性の識別子は、一意の OID にコード化されます。DMI テーブルのオブジェクトにアクセスしたり、DMI テーブルの行を識別したりするときは、INDEX 句を使います。GETNEXT の実行中に DMI テーブルの属性を検査するときは、すべての行識別子 (キーの値) を調べることによって、属性の字句解析上の順序を保ちます。

マスターエージェントによる SET 要求の処理は、GET 要求よりも少し複雑です。通常のコマンドシーケンスでは、GET コマンドの次に SET コマンドがマスターエージェントから送信されます。渡された SNMP PDU に対し、SET コマンドを全部完了できなかったときは、別の SET コマンドをサブエージェントに送信します。この

SET コマンドには、GET コマンドから得られた `oid` 値が指定されます。サブエージェントは SET コマンドを受信すると、いくつかの基本的なチェックを行います。基本的なチェックとは、オブジェクトとインスタンスの存在、値のデータ型、有効な内容、操作に必要とされるメモリーの有効性のチェックなどです。この時点で、属性を保持 (RESERVE) するための `DmiSetAttribute()` による DMI の呼び出しが完了します。RESERVE を使うと、SET を実行しなくても SET の妥当性チェックを行うことができます。RESERVE が失敗すると、マスターエージェントに `genErr` が返されます。RESERVE が成功すると、サブエージェントは `DmiSetAttribute()` に SET オプションを指定して実行することによって、実際に属性を設定します。

別のサブエージェントが、その PDU の SET は実行できないということを SNMP エージェントに通知した場合は、SNMP エージェントは既存の値を使ってマップパーに SET を渡します。

例外の報告

マスターエージェントのトラップ PDU を作成し、そのトラップをマスターエージェントに送信することによって、マップパーがマスターエージェントにトラップを報告します。

トラップは、DMI SP から発行される要求外通知メッセージによって発生します。このメッセージはインジケーションの 1 つです。構成要素から送られたインジケーションは、「イベント」と呼ばれます。マップパーは、SP の登録テーブルにエントリを追加することによって、SP に登録したあと、すべてのインジケーションを受け取ります。また、マップパーは、すべての種類のイベントを受け取るためにフィルタの条件を設定します。マップパーはエージェントからの要求をすでに待っているため、このルーチンは個別のスレッドになります。サブエージェントが、トラップと共に送る OID を決めます。インジケーションには、次のようなさまざまな種類のものがあります。

■ DmiDeliverEvent

このインジケーションは登録先構成要素によって生成されます。SP は、SP によって保守されている登録テーブルとフィルタテーブルに有効なエントリがあるすべての MI アプリケーションに、このインジケーションを渡します。snmpXdmid マップパーは、OID 接頭辞と構成要素 ID を照合し、TrapOID を作成するためのイベントを生成します。groupId と属性もイベントの一部です。snmpXdmid マップパーは、OID 接頭辞を除く、イベントのすべての関連情報を使って、TrapOID を生成します。snmpXdmid は、trapID=14 を使って SNMP 固有のトラップを生成します。

■ DmiComponentAdded

新しい構成要素が登録されると、SPによってこのインジケーションが生成されます。新しい構成要素を、既存の .MAP ファイルまたはまったく新しい .MAP ファイルに登録できます。snmpXdmidはこのトラップを受け取ると、マスターエージェントに登録されているすべてのOIDの登録を解除し、すべての .MAP ファイルを読み込み直して、すべてのOIDをマスターエージェントに再登録するプロセスを実行します。特に、.MAP ファイルはマップパーの外部に生成されるため、これらの処理によってサブエージェントの変換テーブルと .MAP ファイルの同期を保つことが必要です。そのあとでマップパーは、trapID=7でSNMP固有のトラップを生成します。

■ DmiComponent-Deleted

既存の構成要素IDがSPから登録解除されると、SPによってこのインジケーションが生成されます。snmpXdmidマップパーは、このインジケーションを受け取ると変換テーブルを変更します。デフォルトでは、マップパーは、trapID=8のSNMPトラップを生成します。

■ DmiLanguageAdded

このインジケーションは、SPによって生成されます。trapID=9のSNMPトラップが生成されます。

■ DmiLanguageDeleted

このインジケーションは、SPによって生成されます。trapID=10のSNMPトラップが生成されます。

■ DmiGroupAdded

新しいグループが構成要素IDに基づいてSPに登録されると、SPによってこのインジケーションが生成されます。その結果、snmpXdmidの変換テーブルが更新されます。trapID=11のSNMPトラップが生成されます。

■ DmiGroupDeleted

グループがSPから登録解除されると、SPによってこのインジケーションが生成されます。その結果、変換テーブルが更新され、trapID=12のSNMPトラップが生成されます。

■ DmiSubscriptionNotice

このインジケーションは、次の2つの状況下でSPによって生成されます。

管理アプリケーションがインジケーションを受信するための登録を行なったときに、警告タイムスタンプに遭遇した場合。フラグがそのインジケーションの警告の程度を示します。trapID=15 の SNMP トラップが生成されます。

管理アプリケーションがインジケーションを受信するための登録を行なったときに、時間切れタイムスタンプに遭遇した場合。trapID=16 の SNMP トラップが生成されます。

サブエージェントの停止

通常、snmpXdmid デーモンは、明示的に停止されたり、メモリー資源不足のような致命的な状況に遭遇したりしないかぎり、永続的に動作します。

MIF から MIB へのマップ

MIF から MIB へのマップでは、MIF から OID を割り当てる必要があります。トランスレータ (miftomib.EXE) によって、MIF 定義から拡張子が .MAP と .MIB のファイルを構築できます。このトランスレータを、.MIB と .MAP ファイルを生成するためのツールとして使うこともできます。また、テキストエディタを使って手動でマップファイルを生成することもできます。/var/dmi/map の下のディレクトリにあるすべての .MAP ファイルは走査され、サブエージェントの変換テーブルに取り込まれます。

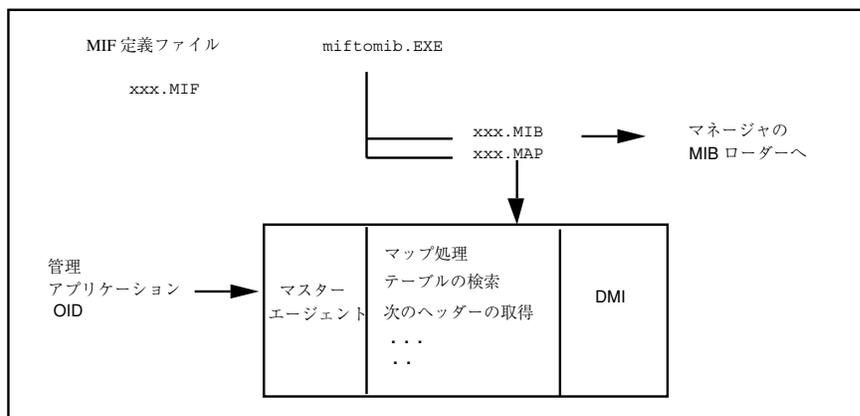


図 6-2 MIF から MIB へのマップ

マップファイルの書式を、表 6-2 に示します。

表 6-2 マップファイルの書式

OID 接頭辞	構成要素名
"1.3.6.1.4.1.42.2000.2"	"Client Access/400 Family - Base Product"

DMI のコンポーネント ID、グループ ID、および属性 ID を MIB のオブジェクト ID にマップすることを目的として設計されています。管理するエンティティでは、マップパーが MIF 定義へのマップに使用する MIB 定義をあらかじめ定義しておく必要があります。miftomib トランスレータを使って MIB マップファイルを生成すると、マップが容易になります。

マップファイルは、マップパーの外部に生成されます。新しい MIF 定義を動的に追加できるようにするには、システムの新しいファイルまたは更新されたファイルについてマップパーに通知するための機構が必要です。これは、マップパーがすべての .MAP ファイルを読み込み直すときに、dmispd によって生成された DmiGroupAdded インジケーションを使うことによって実現されます。

OID 接頭辞と、完成した OID レイアウトの例については、図 6-3 を参照してください。マップパーはマップファイルを使うことによって、OID を SNMP エージェントの中継に登録したり、OID を構成要素名と相互に関連付けたりします。グループに表形式のデータが含まれている場合は、マッピングファイルにコンポーネント ID とグループキーが含まれています。

マップパーでは、OID 接頭辞に制限はありません。OID 接頭辞は、ネットワーク管理者によって制御されます。SNMP マスターエージェントでサブツリーの登録ができるときは、.MAP ファイルにどんな OID を登録してもかまいません。OID の登録に失敗したときは、OID を修正するために .MAP ファイルを変更する必要があります。

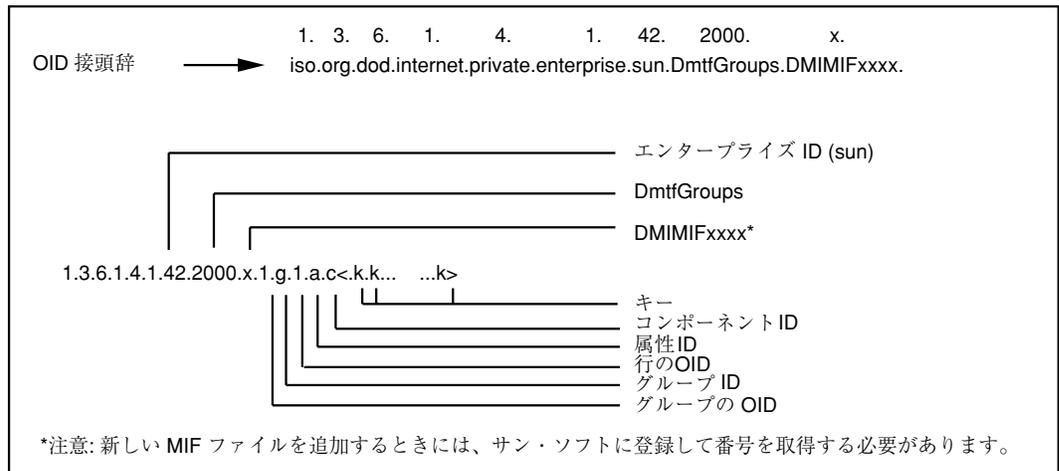


図 6-3 MIB OID レイアウト

オブジェクト識別子を構成する部分を次に示します。

- オブジェクト識別子：OID 接頭辞：1.3.6.1.4.1.42.2000.x
- グループの OID：.1
- テーブルのオブジェクトの種類：.iGroupID
- 行のオブジェクトの種類：.1
- 属性識別子：.iAttributeID
- インスタンス識別子：インデックス：.iComponentID <keys>

クライアントのアクセス属性(キーなし) に対する OID の例を次に示します。

- オブジェクト識別子：OID 接頭辞：1.3.6.1.4.1.42.2000.x
- グループの OID：.1
- テーブルのオブジェクトの種類：.1
- 行のオブジェクトの種類：.1
- 属性識別子：.1
- インスタンス識別子：インデックス：.3

MIB テーブルではカラムからカラムに渡って検索が実行され、MIB テーブルでは行から行に渡って検索が実行されるため、GetNext の操作によってオブジェクトを検索するときは注意が必要です。

特殊なマップの考慮事項

DMI の仕様では、DMI SP に対して ComponentId=1 を予約しています。また、この仕様は、SP の MIF ファイルも定義します。MAP ファイルを作成したり、miftoMIB ユーティリティにのコマンド行パラメタとして OID 設定を指定したりするときは、ネットワーク管理者はこれを考慮する必要があります。すべての MIF ファイルには、ID 1 の標準グループを含める必要があります。

表 6-3 DMI MIB に変換される ComponentID グループ

MIS オブジェクトの識別子と構文	MIF データ	説明	注
DMIcompindex INTEGER (1...217483647)	コンポーネント ID	構成要素の一意の値	SP によってインストール時に割り当てられる。インストールが解除されるまで、SP はこの構成要素とやりとりする。管理アプリケーションは、あとで属性を要求するためにこの ID を記録する
DMIcompManufacture DisplayString (0...64)	属性 「Manufacture」	コンポーネントプロバイダによって割り当てられる値	この構成要素を作成した組織の名前
DMIcompProduct DisplayString (0...64)	属性 「Product」	コンポーネントプロバイダによって割り当てられる値	構成要素の名前
DMIcompVersion DisplayString (0...64)	属性 「Version」	コンポーネントプロバイダによって割り当てられる値	構成要素のバージョン
DMIcompSerialnumber DisplayString (0...64)	属性 「Serial Number」	コンポーネントプロバイダによって割り当てられる値	構成要素のシリアル番号
DMIcompInstallation Date	属性 「Installation」	インストール時に SP によって割り当てられる値	日付と時刻から成る 28 オクテットの表示可能文字列
DMIcompVerify Integer (0...7)	属性 「Verify」	インストール時のこの構成要素の検査レベル	この属性を要求すると、構成要素がまだシステム内にあり、正しく動作しているかどうか調べられる

このマップを使うと、DMI を使ってインストールされる MIF は、管理アプリケーションに表示できる最小限の ComponentID グループを持つだけで済みます。このグループ内の属性はすべて、読み取り専用のアクセス権を持ちます。MIF が基準となって MIB に変換されると、グループ内の属性にアクセス可能となりますが、DmtfGroups ツリーにアンカーポイントが付きます。たとえば、ソフトウェア MIF が定義されていると、変換によって DMISW MIB が作成され、次のようにアンカーポイントが付けられます。

```
enterprise.sun.DmtfGroups.DMISW(2)
```

あるいは、次のようになります。

```
1.3.6.1.4.1.42.2000.3
```

管理アプリケーションでは、同じ構成要素が、DmtfGroups ツリーの 2 つの異なる枝に表示されるように設定されている必要があります。

追加の OID 接頭辞は、次のとおりです。

- DMIHW (3)
- DMIPRINTER (4)

DMI マッパー構成ファイル

デフォルトの構成ファイル `snmpXdmid.conf` は、`/etc/dmi/conf` ディレクトリにあります。この代わりに、コマンド行オプションで、このプログラムに `snmpXdmid.conf` ファイルの位置を表わす別のパスを指定することもできます。

WARNING_TIMESTAMP

`snmpXdmid` は、インジケーションを受け取るために DMI SP に登録する必要があります。DMI 2.0 の仕様では、この登録は、特定のタイムスタンプまで有効です。SP は、登録を終了する前に、登録の警告通知を発行します。このタイムスタンプは、登録の警告通知が発行された時間を示します。

デフォルトの値は、次のとおりです。

```
WARNING_TIMESTAMP = 20101231110000.000000-420
```

EXPIRATION_TIMESTAMP

登録の期限が実際に切れる時刻を示します。登録し直さないかぎり、この時刻よりあとでにインジケーションを受け取ることはありません。デフォルトでは、実際にその登録が永久的に存在するように、かなり先のタイムスタンプが選択されます。この選択が行われると、snmpXdmid は、常にそのインジケーションの登録を SP に保存します。

デフォルトの値は、次のとおりです。

```
EXPIRATION_TIMESTAMP = 250101231120000.000000-420
```

FAILURE_THRESHOLD

DMI SP が snmpXdmid にインジケーションを配信するときにエラーが発生した場合、インジケーションを破棄し登録のエントリをクリアする前に、インジケーションの配信を試みる回数を示します。

デフォルトの値は、次のとおりです。

```
FAILURE_THRESHOLD = 1
```

TRAP_FORWARD_TO_MAGENT

この値がゼロ以外の場合、snmpXdmid は、DMI SP のインジケーションに従った SNMP トラップを生成します。この値がゼロの場合、DMI SP のインジケーションに従った SNMP トラップを生成しません。

デフォルトの値は、次のとおりです。

```
TRAP_FORWARD_TO_MAGENT = 1
```

MIB ファイルの生成

通常、SNMP 管理アプリケーションは、管理データを定義する MIB を必要とします。SNMP マネージャにアクセスできるようにしたい各 MIB ファイルに対して、その MIB ファイルに対応する SNMP MIB を生成します。次に、DMI 構成要素と通信するために MIB の定義を使う管理アプリケーションに、MIB をロードします。管理アプリケーションは、ブラウザや他の MIB ベースのアプリケーションに対して MIB 情報を利用可能にすることもできます。

MIF ファイルから SNMP MIB を生成するには、コマンドプロンプトで `miftomib` ユーティリティを使います。MIB ファイルを作成したら、それを SNMP 管理アプリケーションに転送できます。

DMI コマンド行ユーティリティ

- 75ページの「DMI コマンド行ユーティリティについて」
- 76ページの「dmi_cmd コマンドの使用方法」
- 81ページの「dmiget コマンドの使用方法」

DMI コマンド行ユーティリティについて

デスクトップ管理インタフェース (DMI) プロトコルのコマンド行インタフェースは、DMI サービスプロバイダ (SP) の情報を検索する次の2つのユーティリティで構成されます。

- dmi_cmd
- dmiget

dmi_cmd ユーティリティ

dmi_cmd ユーティリティは、次の機能を提供します。

- DMI SP のバージョン情報を取得する
- 管理アプリケーションが必要とする言語を記述するための構成の設定を行う
- セッションで使用している現在の言語を示す構成情報を取得する
- 構成要素をデータベースにインストールする

- インストールされているものを示すために、システム内の構成要素を一覧表示する
- データベースから既存の構成要素を削除する
- グループスキーマをデータベース内の既存の構成要素にインストールする
- 構成要素内のすべてのグループのクラス名を一覧表示する
- 構成要素内のグループを一覧表示する
- 構成要素からグループを削除する
- データベース内の既存の構成要素の言語スキーマをインストールする
- 指定した構成要素に対してインストールされている言語マップのセットを一覧表示する
- 構成要素の特定の言語マップを削除する
- グループ内の1つ以上の属性のプロパティを一覧表示する

dmiget ユーティリティ

dmiget ユーティリティは、DMI SP 内の特定の構成要素のテーブル情報を検索します。

dmi_cmd コマンドの使用方法

```
dmi_cmd [-s hostname]
-h |
-v |
-w config |
-x |
-CI mif_filename |
-CL [-c compld] [-r reqMode] [-d] [-p] [-m maxCount] |
-CD -c compld |
-GI schema_filename -c compld |
-GL -c compld -g groupld [-r reqMode] [-d] [-p] [-m maxCount] |
-GM -c compld [-m maxCount] |
-GD -c compld -g groupld |
-NI schema_filename -c compld |
-NL -c compld |
-ND -c compld -l language_string |
-AL -c compld -g groupld [-a attrld] [-r reqMode] [-d] [-p]

[-m maxCount]
```

dmi_cmd のコマンド行引数を、表 7-1 に示します

表 7-1 dmi_cmd のコマンド行引数

引数	説明
-s <i>hostname</i>	<i>dmiisp</i> を実行するホストマシンを指定する。デフォルトのホストは、ローカルホスト
-h	使用方法についての情報を表示する
-V	DMI SP についてのバージョン情報を表示する
-W <i>config</i>	<i>config</i> に指定した構成を <i>dmiisp</i> に設定する
-X	使用している現在の言語を示す構成情報を検索する
-CI <i>mif_filename</i>	<i>mif_filename</i> に指定した構成要素をインストールする
-CL [-c <i>compld</i>] [-r <i>reqMode</i>][-d] [-p] [-m <i>maxCount</i>]	構成要素を一覧表示する
-CD -c <i>compld</i>	<i>compld</i> に指定した構成要素を削除する
-GI <i>schema_filename</i> -c <i>compld</i>	<i>schema_filename</i> に指定したグループスキーマをインストールする
-GL -c <i>compld</i> -g <i>groupId</i> [-r <i>reqMode</i>] [-d] [-p] [-m <i>maxCount</i>]	指定した構成要素のグループを一覧表示する
-GM -c <i>compld</i> [-m <i>maxCount</i>]	指定した構成要素のクラス名を一覧表示する
-GD -c <i>compld</i> -g <i>groupId</i>	指定した構成要素の指定したグループを削除する
-NI <i>schema_filename</i> -c <i>compld</i>	<i>schema_filename</i> に指定した言語スキーマをインストールする
-NL -c <i>compld</i>	指定した構成要素の言語マップを一覧表示する

表 7-1 dmi_cmd のコマンド行引数 続く

引数	説明
-ND -c <i>compld</i> -l <i>language_string</i>	指定した構成要素の言語マップを削除する
-AL -c <i>compld</i> -g <i>groupid</i> [-a <i>attrId</i>] [-r <i>reqMode</i>] [-d] [-p] [-m <i>maxCount</i>]	指定した構成要素の属性を一覧表示する

注 - *compld*、*groupid*、*attrId*、および *maxCount* の値は正の整数です。デフォルトの値は 0 です。

reqMode の有効な整数の値は、次のとおりです。

- 1 (DMI_UNIQUE)
- 2 (DMI_FIRST)
- 3 (DMI_NEXT)

注 - *reqMode* のデフォルトの値は 1 (DMI_UNIQUE) です。*reqMode* に無効な値を指定すると、デフォルトの値が使われます。

構成要素 (-CL)、グループ (-GL)、または属性 (-AL) を一覧表示するとき、-d オプションを使うと記述内容が表示されます。また、-p オプションを使うと、プラグマ文字列が表示されます。

dmi_cmd の例

例 1

次のコマンドは、最大 5 個の構成要素のコンポーネント ID、名前、記述を一覧表示し、デフォルトの要求モード (DMI_UNIQUE) を使うことによって、ホスト snowbell で実行している dmispd の構成要素 3 から起動します。

```
% dmi_cmd -s snowbell -CL -d -c 3 -m 5
```

```
Connecting to dmispd on the snowbell...
```

```
CompId: 4
```

```
Comp Name: DMTF Developers - Direct Interface Version
```

Description: A list of the people who actually wrote the code.

CompId: 5
Comp Name: DMTF Developers - Direct Interface Version
Description: A list of the people who actually wrote the code.

CompId: 6
Comp Name: DMTF Developers - Direct Interface Version
Description: A list of the people who actually wrote the code.

CompId: 7
Comp Name: DMTF Developers - Direct Interface Version
Description: A list of the people who actually wrote the code.

CompId: 8
Comp Name: DMTF Developers - Direct Interface Version
Description: A list of the people who actually wrote the code.

例 2

次のコマンドは、デフォルトの要求モード (DMI_UNIQUE) を使うことによって、ローカルホストで実行している dmispd の構成要素 1 のグループ 1 のすべての属性の ID、名前、記憶領域、アクセス様式、データ型、最大サイズを一覧表示します。記述は表示されません。また、最大カウントには制限が設定されていません。

```
% dmi_cmd -AL -g 1 -c 1

Connecting to dmispd on the localhost...

12 attrs listed for group 1 of comp 1

Attr Id: 1
Name: Manufacturer
Storage: MIF_COMMON
Access: MIF_READ_ONLY
Type: MIF_DISPLAYSTRING
maxSize: 64

Attr Id: 2
Name: Product
Storage: MIF_COMMON
Access: MIF_READ_ONLY
Type: MIF_DISPLAYSTRING
maxSize: 64

Attr Id: 3
Name: Version
Storage: MIF_COMMON
Access: MIF_READ_ONLY
Type: MIF_DISPLAYSTRING
maxSize: 64

Attr Id: 4
Name: Serial Number
Storage: MIF_SPECIFIC
```

Access: MIF_READ_ONLY
Type: MIF_DISPLAYSTRING
maxSize: 64

Attr Id: 5
Name: Installation
Storage: MIF_SPECIFIC
Access: MIF_READ_ONLY
Type: MIF_DATE
maxSize: 0

Attr Id: 6
Name: Verify
Storage: MIF_SPECIFIC
Access: MIF_READ_ONLY
Type: MIF_INTEGER
maxSize: 0

Attr Id: 7
Name: ComponentId
Storage: MIF_SPECIFIC
Access: MIF_READ_ONLY
Type: MIF_INTEGER
maxSize: 0

Attr Id: 8
Name: ComponentName
Storage: MIF_SPECIFIC
Access: MIF_READ_ONLY
Type: MIF_DISPLAYSTRING
maxSize: 256

Attr Id: 9
Name: ComponentDesc
Storage: MIF_SPECIFIC
Access: MIF_READ_ONLY
Type: MIF_DISPLAYSTRING
maxSize: 256

Attr Id: 10
Name: GroupId
Storage: MIF_SPECIFIC
Access: MIF_READ_ONLY
Type: MIF_INTEGER
maxSize: 0

Attr Id: 11
Name: GroupName
Storage: MIF_SPECIFIC
Access: MIF_READ_ONLY
Type: MIF_DISPLAYSTRING
maxSize: 256

Attr Id: 12
Name: LanguageName
Storage: MIF_SPECIFIC
Access: MIF_READ_ONLY
Type: MIF_DISPLAYSTRING
maxSize: 256

例 3

次のコマンドは、localhost で実行中の dmispd の namedir.mif をインストールします。ファイル namedir.mif は、構成ファイルで指定したディレクトリに配置されます。

```
% dmi_cmd -CI namedir.mif
Connecting to dmispd on the localhost...
"namedir.mif" is installed as comp 21.
```

例 4

次のコマンドは、localhost で実行中の dmispd の構成要素 5 のインストールを解除します。

```
% dmi_cmd -CD -c 5
Connecting to dmispd on the localhost...
comp 5 is uninstalled.
```

例 5

次のコマンドは、マシン snowbell で実行中の dmispd のバージョンを表示します。

```
% dmi_cmd -s snowbell -V
Connecting to dmispd on the snowbell...
dmispd version: Dmi2.0
description: This is a DMI2.0 based on ONC RPC
```

dmiget コマンドの使用方法

```
dmiget [-s hostname]
-h |
{-c compld [-g groupld] [-a attrld] }
```

dmiget のコマンド行引数について、表 7-2 に説明します。

表 7-2 dmiget のコマンド行引数

引数	説明
<code>-s hostname</code>	dmiget を実行しているホストマシンを指定する。デフォルトのホストはローカルホスト
<code>-h</code>	使用方法についての情報を表示する
<code>-c compId</code>	指定した構成要素のすべてのテーブル情報を表示する
<code>-g groupId</code>	<code>-c</code> 引数を使って指定した構成要素のグループ情報を表示する
<code>-a attrId</code>	<code>-c</code> 引数を使って指定した構成要素の属性を表示する

dmiget の例

例 1

次のコマンドは、構成要素 3 のグループ 2 のテーブル情報を表示します。

```
% dmiget -c 3 -g 2
Connecting to dmspd on the localhost...
For group 2 of component 3:
Id: 10, 10
Id: 20, developer1
Id: 30, SunSoft
Id: 40, Solaris 2.6

Id: 10, 20
Id: 20, developer2
Id: 30, SunSoft
Id: 40, Solaris 2.6

Id: 10, 30
Id: 20, developer3
Id: 30, SunSoft
Id: 40, Solaris 2.6
```

例 2

次のコマンドは、構成要素 3 のテーブル情報を表示します。

```
% dmiget -c 3

Connecting to dmispd on the localhost...

For group 1 of component 3:

Id: 1, SunSoft
Id: 2, DMTF Demonstration
Id: 3, Version 1.0
Id: 4, 1.00000
Id: 5, 1994 06 03 09 00 00
Id: 6, 0
Id: 7, 0
Id: 8,
Id: 9,
Id: 10, 0
Id: 11,
Id: 12,

For group 2 of component 3:

Id: 10, 10
Id: 20, developer1
Id: 30, SunSoft
Id: 40, Solaris 2.6

Id: 10, 20
Id: 20, developer2
Id: 30, SunSoft
Id: 40, Solaris 2.6

Id: 10, 30
Id: 20, developer3
Id: 30, SunSoft
Id: 40, Solaris 2.6

For group 42 of component 3:

Id: 1, Circus
Id: 2, 4.0a

Id: 1, Disk Blaster
Id: 2, 2.0c

Id: 1, Oleo
Id: 2, 3.0

Id: 1, Presenter
Id: 2, 1.2
```

例 3

次のコマンドは、構成要素 3 のグループ 2 の属性 20 のテーブル情報を表示します。

```
% dmiget -c 3 -g 2 -a 20 -s snowbell
```

Connecting to dmispd on the snowbell...

For group 2 of component 3:

Id: 20, developer1

Id: 20, developer2

Id: 20, developer3

エラーメッセージ

- 85ページの「メッセージの名称」
- 94ページの「メッセージの数値表現」

メッセージの名称

この節では、Solstice Enterprise Agents (SEA) の一部のエラーメッセージを一覧に示します。これらのエラーメッセージはわかりやすく記述されています。また一覧では、メッセージを探しやすくするために、メッセージをアルファベット順に示します。リストの右側には、メッセージの番号を示します。

メッセージの番号順に示したメッセージの一覧は、94ページの「メッセージの数値表現」を参照してください。

追加メッセージ

ADD_XLATE_outOfMemory

738

AR 接続メッセージ

AR_CONNECT_AND_OPEN_connectFailed	102
AR_CONNECT_AND_OPEN_getHostNameFailed	101
AR_CONNECT_AND_OPEN_openFailed	103

すべてのエラーメッセージは、Syslog の機能により /var/adm/messages にログが記録されます。

AR および DMI 登録メッセージ

AR_REGISTER_noneRegistered	112
AR_REGISTER_someRegistered	111
DMI_REGISTER_badCmdHandle	833
DMI_REGISTER_badLevelCheck	832
DMI_REGISTER_failed	831
DMI_REGISTER_outOfMemory	838

構築メッセージ

BUILD_OID_LIST_outOfMemory	748
BUILD_SET_ATTRIBUTE_failed	960
BUILD_SET_ATTRIBUTE_outOfMemory	968

クローズメッセージ

AR_CLOSE_failed	
-----------------	--

作成メッセージ

CREAT_BUFFER_invalidOid	711
CREAT_BUFFER_outOfMemory	718
CREATE_COMP_LIST_dmiBroke	756
CREATE_COMP_LIST_outOfMemory	758
CREATE_GC_PAIRS_dmiBroke	766
CREATE_GC_PAIRS_outOfMemory	768
実行メッセージ	
DO_DMI_SETUP_createSemFailed	801
DO_DMI_SETUP_createQueueFailed	802
抽出メッセージ	
EXTRACT_OID_invalidOid	721
EXTRACT_OID_outOfMemory	728
検索メッセージ	
FIND_ATTRIBUTE_ACCESS_notFound	981
FIND_ATTRIBUTE_ACCESS_otherError	982
FIND_ATTRIBUTE_ACCESS_dmiBroke	986
FIND_ATTRIBUTE_TYPE_notFound	991
FIND_ATTRIBUTE_TYPE_otherError	992
FIND_ATTRIBUTE_TYPE_dmiBroke	992

取得メッセージ

GET_ATTRIBUTE_arBroke	247
GET_ATTRIBUTE_cnfBufAddressability	241
GET_ATTRIBUTE_dmiBroke	246
GET_ATTRIBUTE_genErr	245
GET_ATTRIBUTE_tooBig	254
GET_COMP_MIB_arBroke	257
GET_COMP_MIB_dmiBroke	256
GET_COMP_MIB_genErr	255
GET_COMP_MIB_noSuchInstance	251
GET_COMP_MIB_noSuchInstanceOrObject	252
GET_COMP_MIB_noSuchObject	253
GET_COMP_MIB_tooBig	254
GET_DMI_KEY_SIZE_badAttributeType	431
GET_FIND_GC_PAIR_pairNotFound_groupMatch	221
GET_FIND_GC_PAIR_pairNotFound_noGroupMatch	222
GET_GC_PAIRS_dmiBroke	776
GET_GC_PAIRS_outOfMemory	778
GET_INTERNAL_KEY_SIZE_otherError	441
GET_NEXT_COMP_MIB_genErr	405
GET_NEXT_COMP_MIB_outOfMemory	408

GET_NEXT_COMP_MIB_tooBig	401
GET_KEY_VALUE_noneHigher	411
GET_KEY_VALUE_otherError	412
GET_KEY_VALUE_outOfMemory	418
GET_PARSE_INSTANCE_notFullySpecified	211
GET_PARSE_INSTANCE_programmingError	219

GN メッセージ

GN_FIND_COMPONENT_notFound	363
GN_FIND_GROUP_foundNoChange	341
GN_FIND_GROUP_notFound	342
GN_FIND_KEY_noKeyExists	371
GN_FIND_KEY_notFound	372
GN_GET_OBJECT_dmiBroke	386
GN_GET_OBJECT_genErr	385
GN_GET_OBJECT_outOfMemory	388
GN_GET_OBJECT_tooBig	381
GN_PARSE_INSTANCE_badOid	321

初期化メッセージ

INIT_SUB_AGENT_createSemFailed	91
INIT_SUB_AGENT_outOfMemory	98
INIT_SUB_AGENT_rpcregfailed	90
発行メッセージ	
ISSUE_LIST_COMP_badCmdHandle	902
ISSUE_LIST_COMP_badLevelCheck	901
ISSUE_LIST_COMP_failed	900
ISSUE_LIST_COMP_outOfMemory	908
ISSUE_LIST_GROUP_badCmdHandle	912
ISSUE_LIST_GROUP_badLevelCheck	911
ISSUE_LIST_GROUP_failed	910
ISSUE_LIST_GROUP_outOfMemory	918
ISSUE_LIST_ATTRIBUTE_badCmdHandle	922
ISSUE_LIST_ATTRIBUTE_badLevelCheck	921
ISSUE_LIST_ATTRIBUTE_failed	920
ISSUE_LIST_ATTRIBUTE_outOfMemory	928
ISSUE_LIST_DESCRIPTION_badCmdHandle	932
ISSUE_LIST_DESCRIPTION_badLevelCheck	931
ISSUE_LIST_DESCRIPTION_failed	930
ISSUE_LIST_DESCRIPTION_outOfMemory	938

ISSUE_GET_ATTRIBUTE_badCmdHandle	942
ISSUE_GET_ATTRIBUTE_badLevelCheck	941
ISSUE_GET_ATTRIBUTE_failed	940
ISSUE_GET_ATTRIBUTE_outOfMemory	948
ISSUE_GET_ROW_badCmdHandle	952
ISSUE_GET_ROW_badLevelCheck	951
ISSUE_GET_ROW_failed	950
ISSUE_GET_ROW_outOfMemory	958
ISSUE_SET_ATTRIBUTE_badCmdHandle	972
ISSUE_SET_ATTRIBUTE_badLevelCheck	971
ISSUE_SET_ATTRIBUTE_failed	970
ISSUE_SET_ATTRIBUTE_outOfMemory	
キーメッセージ	
KEY_FROM_AR_illegalKey	873
KEY_FROM_AR_otherError	874
KEY_FROM_AR_outOfMemory	878
KEY_FROM_AR_tooLong	872
KEY_FROM_AR_tooShort	871
KEY_TO_AR_otherError	881
KEY_TO_AR_outOfMemory	888

準備メッセージ

PREPARE_FOR_DMI_INVOKE_resetSemFailed 821

設定メッセージ

SET_ADD_TO_RESERVE_LIST_outOfMemory 618

SET_ATTRIBUTE_badPacketType 552

SET_ATTRIBUTE_dmiBroke 556

SET_ATTRIBUTE_genErr 555

SET_ATTRIBUTE_otherError 551

SET_CHECK_ILLEGAL_dmiBroke 576

SET_CHECK_ILLEGAL_genErr 575

SET_CHECK_ILLEGAL_gotError 571

SET_COMP_MIB_badPacketType 602

SET_COMP_MIB_commitFailed 581

SET_COMP_MIB_dmiBroke 606

SET_COMP_MIB_genErr 605

SET_COMP_MIB_noAccess 583

SET_COMP_MIB_noCreation 584

SET_COMP_MIB_notWritable 591

SET_COMP_MIB_resourceUnavailable 601

SET_COMP_MIB_wrongLength 593

SET_COMP_MIB_wrongType	592
SET_COMP_MIB_wrongValue	594
トレースメッセージ	
TRACE_KEY_outOfMemory	1008
変換メッセージ	
TRANSLATE_DMI_KEY_TO_INTERNAL_otherError	421
TRANSLATE_DMI_KEY_TO_INTERNAL_outOfMemory	428
DMI 登録解除および登録解除メッセージ	
DMI_UNREGISTER_failed	841
DMI_UNREGISTER_outOfMemory	848
UNREG_BY_AR_failed	171
XLATE メッセージ	
XLATE_TYPE_mifUnknownType	893
XLATE_TYPE_noError_opaqueType	891
XLATE_TYPE_otherError	894
XLATE_TYPE_unexpectedType	892

メッセージの数値表現

90	INIT_SUB_AGENT_rpcregfailed
91	INIT_SUB_AGENT_createSemFailed
98	INIT_SUB_AGENT_outOfMemory
101	AR_CONNECT_AND_OPEN_getHostNameFailed
102	AR_CONNECT_AND_OPEN_connectFailed
103	AR_CONNECT_AND_OPEN_openFailed
111	AR_REGISTER_someRegistered
112	AR_REGISTER_noneRegistered
171	UNREG_BY_AR_failed
211	GET_PARSE_INSTANCE_notFullySpecified
219	GET_PARSE_INSTANCE_programmingError
221	GET_FIND_GC_PAIR_pairNotFound_groupMatch
222	GET_FIND_GC_PAIR_pairNotFound_noGroupMatch
241	GET_ATTRIBUTE_cnfBufAddressability
244	GET_ATTRIBUTE_tooBig
245	GET_ATTRIBUTE_genErr
246	GET_ATTRIBUTE_dmiBroke
247	GET_ATTRIBUTE_arBroke

251 GET_COMP_MIB_noSuchInstance
252 GET_COMP_MIB_noSuchInstanceOrObject
253 GET_COMP_MIB_noSuchObject
254 GET_COMP_MIB_tooBig
255 GET_COMP_MIB_genErr
256 GET_COMP_MIB_dmiBroke
257 GET_COMP_MIB_arBroke
321 GN_PARSE_INSTANCE_badOid
341 GN_FIND_GROUP_foundNoChange
342 GN_FIND_GROUP_notFound
363 GN_FIND_COMPONENT_notFound
371 GN_FIND_KEY_noKeyExists
372 GN_FIND_KEY_notFound
385 GN_GET_OBJECT_genErr
386 GN_GET_OBJECT_dmiBroke
388 GN_GET_OBJECT_outOfMemory
401 GET_NEXT_COMP_MIB_tooBig
405 GET_NEXT_COMP_MIB_genErr
408 GET_NEXT_COMP_MIB_outOfMemory
411 GET_KEY_VALUE_noneHigher

412 GET_KEY_VALUE_otherError
418 GET_KEY_VALUE_outOfMemory
421 TRANSLATE_DMI_KEY_TO_INTERNAL_otherError
428 TRANSLATE_DMI_KEY_TO_INTERNAL_outOfMemory
431 GET_DMI_KEY_SIZE_badAttributeType
441 GET_INTERNAL_KEY_SIZE_otherError
552 SET_ATTRIBUTE_badPacketType
555 SET_ATTRIBUTE_genErr
556 SET_ATTRIBUTE_dmiBroke
571 SET_CHECK_ILLEGAL_gotError
575 SET_CHECK_ILLEGAL_genErr
576 SET_CHECK_ILLEGAL_dmiBroke
581 SET_COMP_MIB_commitFailed
583 SET_COMP_MIB_noAccess
584 SET_COMP_MIB_noCreation
591 SET_COMP_MIB_notWritable
592 SET_COMP_MIB_wrongType
593 SET_COMP_MIB_wrongLength
594 SET_COMP_MIB_wrongValue
601 SET_COMP_MIB_resourceUnavailable

602 SET_COMP_MIB_badPacketType
605 SET_COMP_MIB_genErr
606 SET_COMP_MIB_dmiBroke
618 SET_ADD_TO_RESERVE_LIST_outOfMemory
711 CREAT_BUFFER_invalidOid
718 CREAT_BUFFER_outOfMemory
721 EXTRACT_OID_invalidOid
728 EXTRACT_OID_outOfMemory
738 ADD_XLATE_outOfMemory
756 CREATE_COMP_LIST_dmiBroke
758 CREATE_COMP_LIST_outOfMemory
766 CREATE_GC_PAIRS_dmiBroke
768 CREATE_GC_PAIRS_outOfMemory
776 GET_GC_PAIRS_dmiBroke
778 GET_GC_PAIRS_outOfMemory
821 PREPARE_FOR_DMI_INVOKE_resetSemFailed
833 DMI_REGISTER_badCmdHandle
841 DMI_UNREGISTER_failed
848 DMI_UNREGISTER_outOfMemory
871 KEY_FROM_AR_tooShort

872 KEY_FROM_AR_tooLong
873 KEY_FROM_AR_illegalKey
874 KEY_FROM_AR_otherError
878 KEY_FROM_AR_outOfMemory
891 XLATE_TYPE_noError_opaqueType
892 XLATE_TYPE_unexpectedType
893 XLATE_TYPE_mifUnknownType
894 XLATE_TYPE_otherError
900 ISSUE_LIST_COMP_failed
901 ISSUE_LIST_COMP_badLevelCheck
902 ISSUE_LIST_COMP_badCmdHandle
908 ISSUE_LIST_COMP_outOfMemory
910 ISSUE_LIST_GROUP_failed
911 ISSUE_LIST_GROUP_badLevelCheck
912 ISSUE_LIST_GROUP_badCmdHandle
918 ISSUE_LIST_GROUP_outOfMemory
920 ISSUE_LIST_ATTRIBUTE_failed
921 ISSUE_LIST_ATTRIBUTE_badLevelCheck
922 ISSUE_LIST_ATTRIBUTE_badCmdHandle
928 ISSUE_LIST_ATTRIBUTE_outOfMemory

930 ISSUE_LIST_DESCRIPTION_failed
931 ISSUE_LIST_DESCRIPTION_badLevelCheck
932 ISSUE_LIST_DESCRIPTION_badCmdHandle
938 ISSUE_LIST_DESCRIPTION_outOfMemory
940 ISSUE_GET_ATTRIBUTE_failed
941 ISSUE_GET_ATTRIBUTE_badLevelCheck
942 ISSUE_GET_ATTRIBUTE_badCmdHandle
948 ISSUE_GET_ATTRIBUTE_outOfMemory
950 ISSUE_GET_ROW_failed
951 ISSUE_GET_ROW_badLevelCheck
952 ISSUE_GET_ROW_badCmdHandle
958 ISSUE_GET_ROW_outOfMemory
960 BUILD_SET_ATTRIBUTE_failed
968 BUILD_SET_ATTRIBUTE_outOfMemory
970 ISSUE_SET_ATTRIBUTE_failed
971 ISSUE_SET_ATTRIBUTE_badLevelCheck
972 ISSUE_SET_ATTRIBUTE_badCmdHandle
978 ISSUE_SET_ATTRIBUTE_outOfMemory
981 FIND_ATTRIBUTE_ACCESS_notFound
982 FIND_ATTRIBUTE_ACCESS_otherError

986	FIND_ATTRIBUTE_ACCESS_dmiBroke
991	FIND_ATTRIBUTE_TYPE_notFound
992	FIND_ATTRIBUTE_TYPE_otherError
996	FIND_ATTRIBUTE_TYPE_dmiBroke
1008	TRACE_KEY_outOfMemory

用語集

この用語集では、Solstice Enterprise Agents のマニュアルの中で使われている専門用語、または Solstice 環境で固有の意味を持つ用語について簡単に説明します。

API	アプリケーションプログラミングインタフェース。API は、製品の機能にアクセスしたり、その機能を使用してアプリケーションを開発するためのソフトウェアルーチンの集合。
ARP	Address Resolution Protocol (アドレス解決プロトコル)。インターネットアドレスに対応するネットワークハードウェアのアドレスを見つけるためのプロシージャ (RFC 826)。
ASN.1	Abstract Syntax Notation One。ネットワーク管理プロトコルで解釈される仕様のひとつで、機器上で動作するエージェントとマネージャとの間の通信情報を、ネットワークに依存しない様式で符号化するために利用される。
CI	Component Interface (コンポーネントインタフェース)。管理情報へのアクセスを記述することにより、その構成要素が管理可能となる。
DCE	分散型コンピューティング環境。DCE は、OSF (Open System Foundation) によって提供されている。DCE によってクライアントサーバーアーキテクチャに基づいたアプリケーションの開発が可能となる。
DMI	デスクトップ管理インタフェース。デスクトップ管理インタフェースは、システム内に存在する管理アプリケーションと構成要素とを仲介するインタフェースとサービスプロバイダの集まりである。

DMI は、特定のオペレーティングシステムまたは管理プロセスと結び付きを持たない独立型インタフェース。

DMTF	Data Management Task Force (データマネジメントタスクフォース)。デスクトップマネジメントタスクフォースは、Digital Equipment Corporation、Hewlett-Packard、IBM、Intel、Microsoft、Novell、Sun、SynOptics の 8 社が共同して 1992 年 5 月に設立された。
IETF	Internet Engineering Task Force (インターネット特別技術調査委員会)。MIB、SNMP のソース。
IP アドレス	TCP/IP を基盤とするインターネットの接続点を表すために使用される 32 ビットの符号。
ISO	International Standard Organization (国際標準化機構)。技術的に幅広い領域に渡り、国際的な取り決めに従って規格の開発を行っている。
MetaData (メタデータ)	ネットワーク内の被管理オブジェクトを記述するために使用されるデータのフォームについての記述の集合。データそのものとは区別される。
MIB	Management Information Base (管理情報ベース)。ネットワーク内部の資源に関する情報を分類するためのもので、階層構造を持つ。業界の合意に基づき、個々の製造業者毎にツリー構造上の特定の位置が割り当てられ、そこに製造した装置に特有な記述を付加することができる。
MIB モジュール	管理されているオブジェクトの集まり。
NMF	Network Management Forum。OSI 技術を使用した相互運用が可能なネットワーク管理の普及を目的とした、ネットワーク機器とソフトウェアのベンダーや開発者の共同組織。
OID	オブジェクト識別子。グローバルオブジェクト登録ツリー内でオブジェクトの位置を識別するための数字。たとえば、ios.org.dod.internet.private.enterprise.synoptics.1.3.2 に相当するのは、1.3.6.1.4.1.45.1.3.2 で、Synoptics3000 コンセントレータとして

識別される。Cisco ルータには `cisco` のように、オブジェクト識別子に MIB の名前を付けることもできる。[S] CMIP では、相対的識別名 (RDN) の組の片方を利用して、「MIT」内のオブジェクトの位置を識別する。[C] Object IDentifer は、標準的ツリー構造の記述を参照して、オブジェクトのクラスを記述するシステムで使用される。ツリーの各ノードには、オブジェクトの識別子が連続した数字になるように数字が割り当てられる。インターネットでは、識別子は、0.128.45.12 のようにドットで区切られた数字文字列として表現して使用される。OSI と Solstice EM では、{ 0 128 45 12 } のように数字を空白で区切り、数字の並び全体を中括弧で囲む。

OID の範囲	サブツリーに含まれている OID の範囲。たとえば、サブツリー 1.2.3 の場合、該当する範囲は 1.2.3 以降となるが、1.2.4 以降は含まれない。
ONC/RPC	オープンネットワークコンピューティング / 遠隔手続き呼び出し。
OSF	Open System Foundation。1988 年に設立。Hewlett-Packard、IBM、および DEC を含む UNIX のコンソーシアム。DME のスポンサーでもある。
OSI	Open Systems Interconnection (開放型システム相互接続)。国際標準化機構 (ISO) によって採択されたネットワーク管理規則の総称。いろいろなメーカーや技術に基づいて製造されたコンピュータの間で通信を容易にするための国際的な取り決め。ISO がこの策定にあたった。
OSI/NMF	OSI Network Management Forum。SNMP、PING、および CMIP プロトコルの定義および規格を策定し、公布するために設立された OSI グループ。
RFC	Request for Comment。TCP/IP ベースのインターネットコミュニティ内でのプロトコルを公式なものとするために記述される一連の文書は RFC とみなされる。文書公開前に正式に標準化されるプロセスの最終段階。RFC は、インターネット特別技術調査委員会 (IETF) によって発行される。
SAP	Service Access Point (サービスアクセスポイント)。サービスユーザーとそのサービスを提供する層の実体 (エンティティ) との接触

点で、そこから個々のユーザーに対して層からサービスが提供される。

- SMI** Structure of Management Information (管理情報構造体)。
- SNM** SunNet Manager。サブエージェントを Site/SunNet/Domain Manager (SNM) に対してエクスポートする場合、SNM スキーマファイルが必要となる。
- SNMP** Simple Network Management Protocol (簡易ネットワーク管理プロトコル)。ネットワークマネージャと種々の被管理オブジェクト内部で動作している「エージェント」との間で情報を交換するためのプロトコル。エージェントは要求に応じてオブジェクトの状態を報告することができる。このプロトコルは、簡易な暫定的解法として導入されたものだが、現在ではインターネット環境で広範囲に渡って利用されている。コネクションレス型のプロトコルのため、ネットワークのパフォーマンスが低下してコネクション指向の高信頼型トランスポートがうまく動作しないような状態になっても被管理オブジェクトからの情報を途絶えることなく受信することができる。
- SNMPD** Simple Network Management Protocol のためのデーモン。
- TCP/IP** Transmission Control Protocol/Internet Protocol。インターネットのプロトコル集は共通のフレームワークに関連したプロトコルを集めたものであったり、典型的には、大規模な通信基盤上で他の開放型(非独占的な)システムとの間でどのようにコンピュータ間の相互通信を行うのかを定めた規約集であったりする。
- UDP** Universal Datagram Protocol。コネクションレスなプロトコルで、SNMP はこの上に実装される。
- イベント** イベントとは、構成要素からサービスプロバイダに対して送り付けられる要求外通知情報 (unsolicited information) から成り、環境の異常で重大な事件や事故の情報を詳細に伝える。イベントが引き金となって、サービスプロバイダは管理アプリケーションに対してインジケーションが送信される。エラーが発生したり、新しいバージョンのソフトウェアをインストールした時に、イベントが送信さ

れることがある。構成要素の製造元が、その製品に関連するイベントと、どの情報がそのイベントとして送信されるのかを定義している。

インジケーション	インジケーションとは、イベントがサービスプロバイダによって受け取られたり、構成要素が MIF データベースにインストールされたりそこから削除された場合に、サービスプロバイダから管理アプリケーションに送られる情報のこと。イベントが原因となるインジケーションの場合、イベントやイベントを送る構成要素についての情報が取められている。
インスタンス	C++ では、クラスのメンバシップによって記述された構造を持つデータのこと。データへのアクセス方法は、クラスで定義されているメンバー関数だけが提供できる。管理されているオブジェクトの場合は、管理されているオブジェクトの特定の筐体や実体のことを指す。たとえば、ルータは 1 つのオブジェクトとみなされ、ある特定のルータはそのクラスのインスタンスである。
インストルメンテーション	インストルメンテーションとは、MIF データベース内の属性に対する値を提供するプログラムの総称。インストルメンテーションは、実行時プログラムによる方法と、インタフェースを直接利用する方法の 2 種類で提供される。実行時プログラムとは、サービスプロバイダによって実行されるプログラムである。このプログラムによって、管理アプリケーションからアクションが要求されたときの値を検索したり、設定したりする。インタフェースを直接利用する方法では、常に稼動状態で、しかもサービスプロバイダにリンクされているプログラムを使うことによって、該当する値を提供する。
インターネット	相互に接続されたネットワークをひとまとまりとした巨大な集合体で、最初は米国内のネットワークで、インターネットプロトコルが動作した。「インターネット」という用語は通常 TCP/IP インターネットワークの集合体のことを指す。
エージェント	ネットワーク管理エージェント (Network Management Agent) とも呼ぶ。ネットワーク上で管理されている資源に常駐するモジュール。資源の状態を報告したり、資源に対する問い合わせに回答したりできる。規約文書 X.701 ISO/IEC 10040 で記述される。一般的には、管理されているオブジェクトで動作するソフトウェアであ

り、そのオブジェクトに関する現在の情報を使って管理アプリケーションに応答やレポートを行う。

エージェント / サブエージェント SDK	ソフトウェア開発キット (Software Development Kit) は、エージェント / サブエージェントライブラリ、MIB コンパイラ、サブエージェントのサンプルなど多数のユーティリティツールを提供する。
エンティティ	システム、構成要素、およびアプリケーションのこと。
親	(子) オブジェクトを含むクラスのインスタンス。[C]
管理アプリケーション	管理可能な製品についての情報をデスクトップシステムで検索したり変更したりするためのプログラム。管理アプリケーションは、マネジメントインタフェース (MI) を介してサービスプロバイダと対話する。リモートネットワーク監視ツールとローカルコントロールパネルは、管理アプリケーションになる。
管理システム	ネットワーク管理システムが動作しているネットワークノードから情報を要求したり、ネットワークノードの情報を設定したりするシステム。
キー	キー属性は、特定のグループ内に属性の集まりの 1 つ以上のインスタンスがあるときに、属性のテーブル内の特定の行を検索するために使用される属性である。たとえば、コンピュータシステムには、通常、1 つ以上のシリアルポートが用意されているが、このシリアルポートを記述する際は、システムの MIF ファイル内のシリアルポートグループをテーブルとして設定し、そのテーブルの 1 つの行に特定のシリアルポートの特性を記述する。この情報にアクセスするときは、I/O アドレスなどの 1 つ以上の属性をキーとして指定する。特定のシリアルポートを検索する場合は、管理アプリケーションは、該当する I/O アドレスを含む行を要求する。
兄弟 (sibling)	当該オブジェクトと共通の親クラスから派生しているオブジェクト。[C]
共通グループ	DMTF によって提唱され認可された MIF グループ。すべての、またはほとんどの管理可能な製品に適用可能な共通の属性を示すもの。たとえば、現場で交換可能なユニット (FRU) や使用中の状態等がこの共通グループに含まれる。

クラス	オブジェクトの集まりを形式的に言い表したもの。OSI では、類似した「属性」と「ふるまい」を持つオブジェクトをクラスに分類する。C++ では、クラスの「インスタンス」と呼ばれるデータ構造の集合の管理規則と「メソッド」（「メンバー関数」とも呼ばれる）で記述される。
クラスインスタンス	実際のクラスを規定できるように、属性の値を収集してあるインスタンス。たとえば、クラスにルーターへのポート情報が含まれている場合、ルーターのボードと特定のポートに対するポート番号を指定することによってインスタンスを1つ規定することができる。クラスを規定するために利用される情報は「インスタンス識別子」と呼ばれる。関連する用語には、インスタンス文字列、相対式別名（Relative Distinguished Name : RDN）、インデックス、名前付きオブジェクトなどがある。
グループ	グループとは、ある所定の構成要素に関連する属性の集合のこと。DMTF グループは、構成要素レベルだけでなくグループレベルでも標準化した MIF を持つ。
ゲートウェイ	2つのネットワークを連結し、一方から他方へパケットを配送するコンピュータ。ゲートウェイは複数のネットワークインタフェースを持つ。
子	クラスのインスタンスに含まれる従属オブジェクト。そのクラスオブジェクトのすぐ下位に存在する。[C]
合成イベント	被管理マシンのさまざまな状態要素を複合的に分析した結果に基づいて生成されるイベント。
構成要素 (コンポーネント)	ハードウェアやソフトウェアの区別なく、デスクトップシステムやサーバーの一部として、あるいはそれに接続または付加されて使用される製品を指す。たとえば、モデム、プリンタ、ネットワークインタフェースカード、表計算プログラム、オペレーティングシステムなど、すべて構成要素とみなすことができる。
サブエージェント	管理情報へアクセスすることができ、システム内部のさまざまなアプリケーションや構成要素に対する管理機能を提供するプロセス。サブエージェントは SNMP を使用してマスターエージェントと通

信する。サブエージェントは、管理アプリケーションと直接通信することはない。

サブツリー	単一の OID によって表現されるものをいい、MIB 全体、フルインスタンス、または MIB 定義で名前が付けられたサブツリーである場合もある。
サブネット	インターネットの世界で使用されるときには、ネットワークのある論理的な区画のことを指す。OSI の世界では意味的にもう少し制限されており、同一の物理的媒体に接続されているネットワークの一部を指す。
サブネットマスク	32 ビットの数値で、IP アドレス内の物理的ネットワークを識別する部分を示すために使用される。
重要度を示す列挙型	いろいろなイベントの重要度を選択するためにフィルタ処理で利用されるビットマスク。
重複登録	あるサブエージェントによってすでに登録されているサブツリーや、サブツリーを含むサブツリーを、別のエージェントが登録しようとする事。
状態	<p>ある時点における被管理オブジェクトに対する「要求」に関する記述。どの時点でも、要求は、その被管理オブジェクトを反映して要求中に定義されている状態にあるか、もしくは状態間を遷移中であったりする。状態をある容器のようなものとみなし、その中に他の状態への遷移に関する情報が置かれていると考えることもできる。ある状態にある間、その状態に応じたポーリング率に応じた間隔に従って、要求が繰り返され、その状態から別の状態への遷移を引き起こす状態にあるかどうかを調べる。</p> <p>ポーリング率のほかに、状態ごとに「シビリティ (重要度)」というものが関連付けられている。シビリティには名前が付けられ、記述を持っている。どの任意の 2 つの状態の間にも、(一方通行もしくは双方向の) 1 つの遷移がある。その遷移ごとに潜在的に多数の状態が関連付けられている。</p> <p>グラウンド (または初期) 状態と呼ばれる必須の状態が存在する。この状態として要求されているのは、シビリティが「normal (正常)」でなければならないということだけである。ほかの状態については、ユーザーの選択したものが使用される。</p>

相互運用性	複数のシステムが、ユーザーの要求に応じて、公に知られている環境中で特定の機能を利用して通信することができる能力のこと。
属性 (情報)	MIF の構築単位。管理可能な製品または構成要素の特性のひとつひとつを属性として記述する。たとえば、プロセッサチップのクロックスピードは、そのチップの属性のひとつである。関連する属性の集合で、MIF グループが構成される。
ディスパッチ	マスターエージェントから 1 つ以上のサブエージェントに管理要求を送ること。ディスパッチは、登録されたサブツリーを示すマスターエージェントの現在のビューと、明示的に定義されたアルゴリズムに従って行われる。
テーブル	特定のグループに対して 1 組以上の属性がある場合に属性テーブルが使用される。たとえば、コンピュータシステムには、通常、1 つ以上のシリアルポートが用意されているが、これらのシリアルポートを表現するためには、システムの MIF ファイル内のシリアルポートグループをテーブルとして設定し、そのテーブルの 1 つの行に特定のシリアルポートの特性を記述することになる。
テーブル	オブジェクトクラスのインスタンスに対する属性値の集合を示す SNMP 用語。行で属性を表現し、そのカラムでクラスインスタンスを表現する。
登録	MIB の「サブツリー」をこれ以降管理することを、サブエージェントがマスターエージェントに通知すること。
独占的なグループ	特定の製品ベンダーに固有の属性のグループ。DMTF によって提唱されたもので、標準化されたものではない。ベンダーはこのグループを利用することによって、自社製品の差別化を計ったり、競争上の優位性を宣伝することができる。
トラップ	インターネット用語では、ポーリングに応じてではなく自発的に、エージェントが管理 MIS に問題を通知すること。SNMP では、形式的には、7 種類のトラップが定義されている。サブタイプを定義することもできる。OMNIPoint 1 では、「トラップ」よりも「イベントレポート」という用語を使用する。

トラップ指示型ポーリング	管理 MIS の詳細情報要求時に、エージェントがポーリングに続いて単一のトラップインジケーションを報告する場合に使用する混成型の障害報告。管理 MIS がさらに情報を要求するとき、ポーリングによって追跡されるエージェントによって1つのトラップが開始される場合の、障害を報告するハイブリッドフォーム。
二重登録	あるサブエージェントによってすでに登録されているサブツリーとまったく同じサブツリーを、別のエージェントが登録しようとする事。
ネットワーク管理エージェント	ネットワーク管理情報をネットワーク管理ステーションと交換するネットワーク管理プロトコル (プログラム) を実装したもの。
ネットワーク管理プロトコル 被管理オブジェクト	<p>管理情報を送るために使用されるプロトコル。</p> <p>ネットワーク資源、または資源の集合を表す表現。一般に、被管理オブジェクトとは、それが表す資源から選択された属性を表す抽象概念である。被管理オブジェクトは MIS 内に存在し、ほかのところにある資源を表す。被管理オブジェクトは、次のような情報で特徴付けられる。</p> <ul style="list-style-type: none"> ■ 属性 - オブジェクトの境界を明示 ■ 管理操作 - 被管理オブジェクトに対して適用されるもの ■ ふるまい - 管理操作に対する応答 ■ 通知 - 管理オブジェクトで発行するもの <p>MIB や MIT のエントリーは被管理ネットワークのノードやリンクの様相を表現するもので、Solstice EM サービスを利用してそのエントリーに値を設定することもある。MIS はオブジェクトに対してポーリングを行ったり、現在のオブジェクトインスタンスに対する属性値を表示したり変更したりすることによって、オブジェクトの管理を行う。</p>
被管理オブジェクトクラス	被管理オブジェクトの集合を形式的に表現したもの。被管理オブジェクトとは、管理されている資源を表すデータの集合のこと。ITU 勧告 C.701 ISO/IEC 10040 で定義されている。

被管理ノード	Solstice EM MIT に入力されているオブジェクトクラスを持つネットワーク上のネットワークコンピュータ、ルーター、ハブ、その他の装置、およびその上で動作するネットワークエージェント。
被要求グループ	DMI 対応にするために、MIF ファイルに登録する必要がある属性のグループ。現時点では、被要求グループは、ComponentID グループだけである。これは MIF ファイルではグループ 1 になっていなければならない。
標準グループ	MIF グループの 1 つで DMTF によって提唱され、認可されている。プリンタやネットワークインタフェースカード等の同種の製品の、ほぼすべてのものに対して適用可能な属性が記述されている。現在、PC システムに対するものが認可され、ネットワーク情報カードに対するものが提案中であり、プリンタ、サーバー、ソフトウェア、およびモデムに対するものが開発中。
フィルタ	ネットワーク管理コマンドを送るべきオブジェクトを選別するために、属性の集合に対してブール式を使用した試験を行うこと。フィルタ処理を通過することのできたオブジェクトの実体（インスタンス）は管理操作を行う対象となる。CMIS 仕様 (ISO/IEC 9595) で定義されているように、フィルタ処理機能を使用することにより、管理プロトコルの利用により発生するネットワークトラフィックのオーバーヘッドを軽減することができる。このフィルタ機能は、UNIX システムのものと利用の仕方が全く異なる。UNIX システムでは、フィルタはプログラムのひとつで、あるストリームから入力を受け付けて別のストリームに対して処理結果を出力するので、必要に応じて別の処理機能に対してパイプで連結することができる。「NMF (Network Management Fourm)」を参照。
プロトコル	コンピュータが相互に通信するときに使用される規約。プロトコルは、専用言語でもあり、OSI のネットワーク階層のプロシージャでもある。
ポーリング	被管理オブジェクトに対して定期的に送られる MIB または MIT オブジェクトクラスの状態情報要求。Solstice EM Request Designer 経由でネットワークマネージャのポーリング方法を設定することがある。SNMP はポーリング指向で、CMIP はイベント指向の傾向である。

マスターエージェント	Domain Manager、Enterprise Manager、HP Openview などのマネージャと SNMP プロトコルメッセージを交換する、被管理ノード上のエンティティまたはプロセス。
マッパー	DMI 2.0 の技術は、マッパーによって統合される。マッパーはサブエージェントとして機能する。マスターエージェントから要求を受信し、それらを適切な DMI 要求に変換する。これらの DNMI 要求は DMI サービスプロバイダに送信される。マッパーは、DMI サービスプロバイダから応答を受信すると、この応答を SNMP 応答に変換し、それをマスターエージェントを介してマネージャに転送する。
ルーター	経路制御 (ルーティング) とは、パケットを送るための経路を選択するプロセスを指す。ルーターは、そのような選択を行うことが可能なコンピュータのことである。ホストとゲートウェイは両方とも経路制御を行うが、一般的にルーターという用語では、2 つのネットワークを相互に連結する装置のことを指すときに使われる。「ゲートウェイ」を参照。
レガシー SNMP エージェント	SNMP に基づく「エージェント」で、すでに製品としてサンや他の企業から出荷されたものを指す。Solstice Enterprise Agents によってレガシー SNMP エージェントを統合することができる。
列挙型	ある属性に対して指定可能な値の一覧を列挙した型をいい、グローバルに利用される場合も、またローカルでのみ利用されることもある。グローバルな列挙型は名前を持ち、構成要素内の他の属性からも利用されることがある。ローカルな列挙型は名前を持たず、その型を持つ属性のみで利用される。

索引

A

acl グループ 42

C

command 変数 38

ComponentId 70

ComponentID グループ 71

D

DMI

DMI とは 16

DMI の概要 75

DMI の機能 48

DMI 要素 16

アーキテクチャ 49

構成要素の名前とそのデフォルトの位置 25

要求 14

要求の変換 63

DMI API ライブラリ 53

DmiComponent-Deleted 66

DmiComponentAdded 66

DmiDeliver イベント 66

dmiget 75

dmiget ユーティリティ 76

dmiget ユーティリティの例 82

コマンドオプション 81

DmiGroupAdded 66

DmiGroupDeleted 66

DMIHW (3) 71

DmiLanguageAdded 66

DmiLanguageDeleted 66

DMIPRINTER (4) 71

DmiSetAttribute 65

DMI SP 50

モジュール 51

dmispd プロセス 23

DMI SP 呼び出しの引数 53, 64

DmiSubscriptionNotice 67

DMISW MIB 71

dmi_cmd 75

dmi_cmd ユーティリティ 75

dmi_cmd ユーティリティの例 78

コマンドのオプション 76

dmi_cmd ユーティリティの要求モード 78

DMI_UNIQUE 78

DMI サービスプロバイダの起動 52

DMI のソフトウェアのデフォルトの位置 25

DMI ブラウザ 50

DMI マッパー 57

DMI マッパーの起動 55

Solaris での実行 60

構造 61

マップ 60

DMI マッパーの起動 55

DMI マッパーの機能 61
DMTF 48
DmtfGroups ツリー 71

E

environment グループ 37
EXPIRATION_TIMESTAMP 72

F

FAILURE_THRESHOLD 72

G

genErr 64
GETNEXT のテーブルの属性 64
GETNEXT 要求 64
GET RESPONSE 31
GET と GETNEXT 30
Get と Set のさまざまな属性 52
GET 要求 64

I

INDEX 句 64

M

MAP ファイル
MAP ファイルの書式 68, 78
サンプル 67
定義された .MAP ファイル 67
max_agent_time_out 変数 37
MI 51
MIB
サンプル 44
発行 43
mibiisa.rsrc ファイル 36
MIB OID 43, 63
MIB OID レイアウト
グラフィック 69
MIB ファイルの生成 72
miftomib.EXE 67
miftomib トランスレータ 68
miftomib ユーティリティ 73
MIF から MIB へのマップ 67
MIF から MIB までのコンパイラ 53
MIF データベース 51, 52

MIF のインストールと削除 52
MIF ファイル 72

N

name 変数 39

O

OID と構成要素の関係付け 60
OID の修正 68
OID の範囲 15

P

policy 変数 38
poll-interval 37
port 変数 40

R

RC スクリプトファイル 23
registration_file 変数 38
RESERVE コマンド 65
resource グループ 37

S

SET 31
SET 要求 65
SNMP 11
DMI との通信 57
DMI マッパー 59
PDU 42
SNMP の機能 15
構成要素の名前とそのデフォルトの位置 23
スキーマファイル 104
デーモン snmpdx 23
プロトコルメッセージ 13
マスターエージェント 59
要求の変換 63
snmpdX.acl ファイル 60
snmpdx.rsrc ファイル 36
SNMP MIB の生成 72
SNMP OID の登録 29
SNMP PDU 65
snmpXdmid 58

snmpXdmid.conf 71
SNMP エージェントへの登録 60
SNMP と DMI との通信
通信経路 59
SNMP に対する DMI サブエージェント 59
SNMP のソフトウェアのデフォルトの位
置 23
Solstice Enterprise Agents のインストール 19
Solstice Enterprise Agents パッケージの構成
要素 24, 26
SP プロセス 51
subtrees 変数 40
SUNWmibii 21
SUNWsacom 20
SUNWsacom パッケージのスクリプトファイ
ル 23
SUNWsadmi 20
SUNWsasnm 20

T

tables 変数 40
Timeout の値 62
timeout 変数 40
TRAP 31
TRAP_FORWARD_TO_MAGENT 72
trap グループ 43
type 変数 37

U

user 変数 38

W

WARNING_TIMESTAMP 71
watch_dog_time 29
watch_dog_time 変数 40

あ

アクセス
MIF データベースへのアクセス 64
アクセス制御構成ファイルの構文の例 41
アクセス制御リストファイルの変数 42
新しいパッケージの追加 22

い

イベントとインジケーションの登録機能およ
びフィルタ機能 49
イベントの送信 52
インストール手順 22

え

エージェント
エージェントアクセス制御ファイル 41
グループで使われる変数 39
サブエージェントへの接続 62
登録ファイル 38
エージェントアクセス制御ファイル 41
エージェント状態 = ACTIVE 62

お

オブジェクトの識別子の部分 69

か

管理インタフェース 51

き

既存のパッケージの削除 22

く

クライアント接続属性の OID
グラフィック 69
グループモードの要求 30

こ

構成要素の登録 60
構成要素を DMI SP に登録 52
コマンド行引数 32
DMI マッパー 55
コマンド行ユーティリティ
dmispd のバージョンの表示 81
namedir.mif のインストールの例 81
構成要素の一覧表示の例 78
構成要素のインストール削除の例 81
構成要素のグループ情報の一覧表示の
例 82
構成要素の情報の一覧表示の例 82

構成要素の属性情報の一覧表示の例 83
属性の一覧表示の例 79
コマンドの分割 48
コンポーネントインタフェース 51

さ

サブエージェント 14
 アーキテクチャ 31
 エージェントへの接続 62
 サブエージェントの起動 28
 サブエージェントの再初期化 63
 サブエージェントの終了 32
 サブエージェントの初期化 62
 サブエージェントの停止 67
 サブエージェントの登録 29, 62
 サブエージェントの変数 37
 識別子 62
 処理プロセス 61
 存在の確立 32
 ブート時の手動または自動での起動 29
 変換テーブルの構築 62
 保守管理 32
 マスターエージェントによる起動 28
サブエージェント ID 62
サブエージェントのアーキテクチャ 31
サブエージェントのアドレス 63
サブエージェントの起動 28
サブエージェントの再初期化 62, 63
サブエージェントの識別子 62
サブエージェントの初期化 62
サブエージェントのタスク 54
 初期化と再インストール 55
 マッパーの起動 55
サブエージェントの停止 67
サブエージェントの動的な登録方法 29
サブエージェント名 63
サブツリー 15
サブツリーのファイルの例 39

し

重複登録 15
初期化 63

せ

静的な方法 29

そ

ソフトウェア開発ツールキット 14
存在の確立 32

た

タイムアウト機構 29
タイムスタンプ 71

つ

通信
 サブエージェントからマスターエージェントへの通信 29

て

ディスパッチ 15
デスクトップマネジメントタスクフォース (DTMF) 102
デフォルトの構成ファイル 71

と

動的エージェントの終了 32
登録 15
登録ファイル 38
登録ファイルの agents グループ 39
登録ファイルの macros グループ 39
独立型インタフェース 102
トラップ ID 61

に

二重登録 15

ね

ネットワーク 11

は

バインドポリシー 29
パッケージ 19
パッケージの構成要素 24, 26

ふ

プラットフォーム 19

分割モードの要求 30

へ

変換テーブル 62

ほ

ポート 161 29

ホストの名前 43

ま

マスター

environment グループ 37

マスターエージェント

コマンド行引数 33, 64, 77, 78

サブエージェントとの通信 29

状態ファイル 43

マスターエージェントの起動 28

マスターエージェントの機能 27, 28

マスターエージェントの定義 17

役割 17

要求の送信 30

リソース構成ファイル 35

マスターエージェント状態ファイル 43

マスターエージェントの起動 28

マスターエージェントのコマンド行引数 33,
64, 77, 78

マッパー 14, 54

マップ

MIF から MIB へのマップ 67

マップでの考慮 70

マップファイル

マップファイルの生成 61

マップファイルの書式 68

ゆ

ユーザーデータグラムプロトコル (UDP) 27

よ

要求の送信 30

用語 14

OID の範囲 15

サブツリー 15

重複登録 15

デイスパッチ 15

二重登録 15

用語の登録 15

り

リソース構成ファイル 35, 36

れ

例外の報告 65

レガシー SNMP エージェント 14