



# **SNMP Research Incorporated**

**Develop Unified SNMP, XML, CLI, and Web-based  
Management for Embedded Real-Time Systems with  
MIBGuide™**

**A Technical White Paper**

## Table of Contents

Overview.....	2
Introduction to MIBGuide Development Tools.....	3
EMANATE Run-Time Extensible Agent System.....	3
EMANATE/Lite Compile-Time Extensible Agent System.....	5
Extending SNMP Research SNMP Agents with MIBGuide - The Five-Step Method.....	6
RTOS and CPU Support.....	7
EMANATE/Lite Footprint Information.....	8
Sources for More Information.....	8

## Overview

Support for remote management is a clear market requirement for all telecom and datacom networking equipment. This is normally provided by an SNMP agent, although other access mechanisms such as CLI, XML, Web, or other protocols may also need to be supported. In addition to the industry standard management objects defined by MIB-2 and other IETF standards, equipment vendors need to support their private or "enterprise" MIB objects.

SNMP Research offers a suite of powerful tools, including a MIB editor application for easy MIB creation, MIB compilers and source code generators automating much of the development process, tight integration with popular IDEs for coding, and a MIB browser application for testing. These tools have been packaged into a single graphical development environment called MIBGuide. MIBGuide allows the developer to focus on what they know best, which is the architecture and internals of the device being managed, and not worry too much about the intricacies of MIB design or SNMP. In addition to speeding up the development and testing process, MIBGuide allows the developer to instrument the data once, and access the agent with SNMP as well as CLI, XML, and Web without any additional development. In addition, MIBGuide assures field-proven standards compliance and interoperability, and gives a straightforward path for patches and upgrades to evolve with changes to the standards documents.

With MIBGuide, the development engineer's confidence level of completing an interoperable, multi-protocol compliant agent on schedule and under budget is very high. The product comes with extensive documentation and example code, and SNMP Research's exceptional support team can provide assistance and consulting every step of the way.

## Introduction to MIBGuide Development Tools

MIBGuide is designed to take the guesswork out of adding management information into the agent. Unlike many other SDKs or IDEs, MIBGuide assumes you have very limited MIB or SNMP knowledge, and outputs nearly all the code needed for a complete extension. All that is needed is limited C or Java-language skills to fill in the automatically generated stub routines. Access with SNMP, CLI, XML, and/or Web is automatic. With MIBGuide, the development process from creating a MIB to testing the implementation is easier than ever. Also included are step-by-step source code examples which address both simple and advanced management concepts. MIBGuide is included in both the EMANATE run-time extensible agent product, and the EMANATE/Lite compile-time extensible agent product.

Here is what one developer said about our development tools...

*Kudos are due your developers at SNMP Research! I've used other SNMP SDKs, and read documentation for still others. They were all quite unpleasant, to be frank. I always felt that I should just have to compile the MIB, provide the MIB data ("instrumentation"), and provide some smarts for advancing table indices. But, the other packages always required me to do that and much more. Not so with Emanate: there's no rough edges, all the corners have been rounded off. It was so pleasant, that not only was I able to focus on my instrumentation, but I also felt encouraged to flesh out the MIBs (RFC 2248 & 2249), providing instrumentation for some of the more esoteric variables which I otherwise would not have done (and did not do in the past when using other SDKs).*

*Thanks!*

*- Dan Newman, Senior Software Engineer, Sun Microsystems, Inc.*

Because the agent extension API is shielded from the system dependent layer of the agent code, a developer can reuse the same code to port to multiple operating systems. Developers complete their project quickly, while providing an infrastructure to easily support future revisions. Many agent SDKs which appear to be "free" at first glance, really begin to cost time and money when developers have to turn the free code into something useful with limited or very lacking documentation and support. With MIBGuide agent development tools, you get the complete package.

## EMANATE Run-Time Extensible Agent System

### EMANATE: Introduction

The EMANATE Run-time Extensible Agent System (EMANATE: Enhanced Management Agent Through Extensions) is designed to address the problem of multiple SNMP agents on a single platform via a modularly constructed extensible agent. EMANATE agents are constructed in a modular, hierarchical manner and consist of a Master Agent and zero to many Subagents which support various MIBs. The Subagents are permitted to connect and disconnect to the Master Agent at will, at which time their MIBs are added and removed dynamically from the Master Agent.

The Master Agent and Subagents communicate over an asynchronous message passing interface which is optimized for a particular architecture. Furthermore, EMANATE places no restrictions on the order in which the Master Agent and Subagents may be started.

EMANATE provides system independent APIs which developers can use for implementing their MIB in an EMANATE Subagent. This system independent API hides the details of the particular system on which the Subagent is being developed and permits much Subagent code reuse across different architectures. Furthermore, the API provides a clear interface to the system dependent portion of the Subagent code.

## **The Master Agent**

The Master Agent, which is MIB independent but protocol dependent, is for all intents and purposes the SNMP agent. It contains the agent protocol engine for SNMPv1, SNMPv2c, and SNMPv3, but also has the capability to communicate CLI, XML, and HTML requests without any additional coding. The Master Agent is in charge of the authentication, authorization, access control, and privacy mechanisms. Each SNMP request is handled by a unique thread for asynchronous operation. In conclusion, the Master Agent does all of the "hard" work.

In most cases of Subagent development, the developer does not work with the Master Agent but rather concentrates entirely upon development of the Subagent. This means that developers need know very little about SNMP, but can instead concentrate on that which they know --- their application, and how it should be managed.

## **The Subagents**

EMANATE Subagents, in contrast to the Master Agent, are MIB dependent and protocol independent (they make no assumptions about whether SNMP, CLI, XML, Web, or any other protocol is used). Because the Master Agent is responsible for the "difficult" tasks, Subagents are simple and easy to implement and test. Subagents, which are built with the MIBGuide Subagent Development Kit, are very modular and are divided into system dependent and system independent portions. The Subagents consist primarily of method routines conforming to the system independent API. Subagents are essentially a small program which provides access to method routines to access the instrumented data.

The method routines are divided into a system independent part and a system dependent part. The system independent method routines are generated automatically by the MIBGuide MIB compiler and source code generator, and stubs are generated for the system dependent method routines (examples are given below). All of the C data structures, or Java classes, and function prototypes necessary for the Subagent are also generated by these same tools. Of course, code specific to the system or application being managed is not generated.

Much of the Subagent code can be reused in multiple EMANATE environments. That is, if you have developed a Subagent on one architecture, you will be able to reuse all of the system independent code on a different architecture since the API is exactly the same. Additionally, the API is the same as that used in other SNMP Research agent products, so the code developed for Subagents can easily be integrated with a monolithic agent (and vice versa).

## **EMANATE/Lite Compile-Time Extensible Agent System**

EMANATE/Lite is the "traditional" monolithic compile-time extensible SNMP agent. This simply means that in order to add new MIB variables to the agent, it is necessary to have access to the source code and to compile and link in the new extensions to the MIB already supported by the agent.

This software provides the core SNMP protocol engine that should be installed within each SNMP managed element or device. It receives and responds to SNMP queries and commands issued from SNMP management stations. EMANATE/Lite provides access to management information for each of the managed protocol layers within the network element.

EMANATE/Lite can be compiled to support SNMPv1, SNMPv2c, and SNMPv3. This trilingual option preserves the investment in current management technology while providing a smooth transition to SNMPv3-based management.

### **Features**

EMANATE/Lite

1. fully supports SNMPv1, SNMPv2c and SNMPv3. This includes support for security and administration, authentication, authorization, access control, and privacy. Optional support for CLI, XML, and/or Web access available.
2. is designed with modular architecture that eases the task of porting EMANATE/Lite to new platforms.
3. can be easily extended to provide support for additional MIB variables.
4. includes the MIBGuide suite of tools which accelerate development.
5. provides method routines that are protocol independent but MIB dependent.
6. supports all MIB II variables, including read-write variables (sets), that can be supported without altering the kernel or device driver source code. The underlying platforms influence which variables are available.
7. has multi-phase set routines which provide compliant write access to the MIB variables.
8. has a singly threaded Agent that processes one Protocol Data Unit (PDU) at a time (synchronous), first in, first out. The Agent parses the PDU packet, processes it, and creates the response.
9. includes Libraries, Utilities, and MIB Tools (LUM).

## **Extending SNMP Research SNMP Agents with MIBGuide - The Five-Step Method**

It is likely that those who purchase an SNMP agent intend to add their own MIB objects to that agent. This document outlines the procedure to add additional MIB objects to any SNMP agent product from SNMP Research using MIBGuide.

### **Step 1: Design, Define and Write the MIB**

When using standard MIB documents are either not possible, or if the developer needs to craft their own distinct enterprise-specific MIB document, MIBGuide's MIB Editor will save the developer a great deal of time. MIBGuide's MIB Editor assumes the developer has minimal MIB design experience and steps the developer through the task of creating an SMIV2-compliant MIB document according to RFC-2578, RFC2579, RFC2580.

Besides a productivity tool, MIBGuide's MIB Editor will verify that the MIB document is syntactically and semantically correct so that it can be compiled by any standards-compliant MIB compiler.

### **Step 2: Create the Extension**

MIBGuide's MIB compilers and source code generator allow the developer to select multiple options for generating the extension code. An easy to navigate wizard allows the developer to choose features for code generation such as: C or Java APIs, stable storage, advanced table routines including RowStatus and searching algorithms, set/notification method routines, and others. The wizard also allows the developer to automatically generate working code for other protocols such as CLI, XML, and Web to access the agent.

### **Step 3: Create the Instrumentation**

MIB objects can only report information that has been gathered by an application, and the agent extension must have access to that information. For example, it is impossible for an agent to support the TCP family of objects if the operating system kernel does not collect the appropriate TCP statistics. If the kernel gathers the necessary information, it must also be made available to the agent's MIB structures.

It is possible that the developer might need to add instrumentation to support the new MIB objects. In the case of the TCP group, one might have to add counters or gauges to the kernel. This instrumentation will then be accessed by the agent's method routines.

If the application already collects the information modeled in the MIB document, or if the developer is replacing or augmenting a proprietary protocol with SNMP, then this step is greatly simplified and may even be bypassed entirely.

#### Step 4: Complete the Method Routines

MIBGuide's source code generator automatically generates a set of method routine stubs from the MIB document. The method routines are those functions which access the instrumentation.

MIBGuide allows the developer to use an easy and familiar integrated development environment (IDE) to complete the method routines, recompile/link, and debug the code. Most, if not all, the code to be completed is located within well defined method routines. So, the developer only needs to know how to hook their data into the automatically generated data structures. When the code is recompiled back into SNMP Research's EMANATE or EMANATE/Lite agent products, the information is now manageable via SNMP, CLI, XML, Web, and possibly many other protocols!

#### Step 5: Test the Instrumentation and Agent

After creating the instrumentation, completing the method routines, and building the Subagent, the developer will need to test the program extensively. MIBGuide offers an SNMPv1, v2c, and v3-compliant MIB browser for performing SNMP-based testing. Other tools may be necessary for testing protocol accessibility such as XML, CLI, or Web, or for stress testing. After this stage, the five-step method for creating MIB extensions is complete. If the MIB needs to be updated, return to step one.

### RTOS and CPU Support

EMANATE and EMANATE/Lite are supported on the following embedded environments. Additional operating systems for open systems are available.

RTOS	Testbed Processor	Cross-Development OS
Chorus ClassiX 3.1	Intel 80x86	Solaris 2.x
INTEGRITY 2000 v4.0.9a	PowerPC	Windows
Linux (MontaVista, Red Hat, SuSE)	Intel 80x86	Linux
LynxOS 3.0.1	Intel 80x86	LynxOS 3.0.1
Nucleus NET 4.0*	AMD 186 (i80c186)	Windows
ENEAS OSE 4.5.0	Soft Kernel	Solaris 2.x/Windows
pSOS 2.2	Motorola 68030/68040	SunOS 4.1.x
VRTX	Motorola 68030/68040	SunOS 4.1.x/Windows
VxWorks 5.4/Tornado 2.0	Motorola 68030/68040	Solaris 2.x/Windows
Windows CE*	Intel 80x86	Windows

\*Only available for EMANATE/Lite

SNMP Research's source code agents are processor independent. We develop the source code on the specified processor. But we have customers who are successfully using our code on other architectures. Makefile macros provide a convenient way to customize the product to compile on other chipset environments.

## EMANATE/Lite Footprint Information

Naturally, agent image size can vary a good bit, depending on the features and size-saving features you turn on and off at compilation. Here is some size information for EMANATE/Lite on pSOS. These values are taken from the linker maps.

SNMPv1-only agent, with MIB-II support, and some optimization turned on:

- The total image requires about 253k of ROM and about 128 of RAM, but these numbers include not only the agent, but also pSOS, PNA, probe, pHILE, pREPC, pRPC.
- If you look only at the agent parts, you get about 25.9 Kbytes for the agent and about **49.2 Kbytes** total.

Trilingual SNMPv1, SNMPv2c, SNMPv3 agent on pSOS (but without all of the space-saving optimizations, which means it can be shrunk to a certain degree):

- 328K code and 35.5k of initialized data again including agent with MIB-II but also pSOS, pROBE, pNA, pHILE, pREPC, pRPC.
- If you look only at the agent and MIB-II part, you get:  
About 64k decimal of code for the agent plus about 36.5k decimal of code for the libraries for a total of about **100.5k** (decimal) of code.

## Sources for More Information

For further information about this or other of SNMP Research's products, please contact SNMP Research, Inc.

SNMP Research Incorporated  
3001 Kimberlin Heights Rd.  
Knoxville, TN 37920  
U.S.A.  
Tel: +1 865 573 1434  
Fax: +1 865 579 6565  
E-mail: [info@snmp.com](mailto:info@snmp.com)  
[www.snmp.com](http://www.snmp.com)