SMBIOS Overview Document

September 2003

System Management BIOS

This overview document introduces the reader to the Distributed Management Task Force (DMTF) System Management BIOS (SMBIOS) Specification. This note describes the benefits of SMBIOS and explains how to position SMBIOS within a corporate management infrastructure.

Introduction

Continuing the DMTF's mission of leading the development, adoption and interoperability of management standards and initiatives for enterprise and Internet environments, the SMBIOS Reference Specification addresses how motherboard and system vendors present hardware-related management information in a standard format, by extending the BIOS interface on Intel architecture systems. The information is intended to allow generic instrumentation to deliver this data to management applications that use the Desktop Management Interface (DMI), Common information Model (CIM) or direct access (during pre-OS environments). It eliminates the need for error prone operations, such as probing system hardware for presence detection.

The SMBIOS specification was started as a joint effort by BIOS vendors and system manufactures in the 1995 timeframe, and was originally referred to as DMI BIOS. The early focus was on specifying mechanisms to provide system information to DMI instrumentation. The SMBIOS Specification was later adopted by DMTF in 1999, and has been periodically updated to keep enumerations current, clarify the wording of the specification and address new hardware capabilities.

SMBIOS

The SMBIOS specification defines both the structure of the information, as well as the access methods (i.e., how to retrieve the information from the system).

The SMBIOS architecture is defined to provide information to BIOS developers regarding extending BIOS to allow their product's hardware and other system-related information to be accurately determined by users of the defined interfaces. This specification also provides information to developers of management instrumentation regarding translating from the SMBIOS format to the format used by their chosen management technology – whether it is a DMTF technology like CIM or DMI, or another technology like Simple Network Management Protocol (SNMP). To support this translation when DMTF technologies are used, sections of the SMBIOS Specification describe the DMI groups and CIM classes intended to convey the information retrieved from an SMBIOS compatible system.

SMBIOS Structures

SMBIOS defines approximately 40 data structures, representing information on various components and settings for the system. To report the system information to an application or instrumentation provider, an SMBIOS-compliant implantation must (at a minimum) populate the following set of base structures:

- BIOS Information
- System Information
- System Enclosure
- Processor
- Cache
- System Slots
- Physical Memory Array
- Memory Device
- Memory Array Mapped Addresses
- Memory Device Mapped Addresses
- System Boot Information

In addition to the above structures, SMBIOS implementations can also represent additional information about the baseboard, system configuration, jumper settings, BIOS languages supported, memory controllers and modules, on-board devices, ports and connectors, pointing devices, batteries and power supplies. Information regarding controllers and service processors in the system can be reported. The configuration options for such devices (e.g., devices conforming to the DMTF's Alert Specification Format [ASF] or the industry's Intelligent Platform Management Interface [IPMI]) are available in SMBIOS.

SMBIOS defines structures for presenting information about the sensors in a system (e.g., voltage, temperature, current, watchdog, etc.) and the state of these sensors.

The BIOS typically populates the SMBIOS structures at system boot time, and is not in control when the OS is running. Therefore, dynamically changing data is rarely represented in SMBIOS tables. In most implementations, it is common practice to use direct interfaces to the management controllers to get access to their dynamic information. The same issue applies to hot-plug components (e.g., PCI hot-plug devices), which may not appear in SMBIOS tables. Alternate system interfaces should be used in such instances.