

SNMP Packet Library

Logical View Report

C++ Syntax

Includes Attributes And Operations

Includes Documentation

Generated January 15, 2003

11:27:32 PM by Rational Rose

ABOUT THIS GUIDE

Audience

This document is intended for software developers who will develop applications using the SNMP Packet Library.

Software License

A software license was furnished prior to the receipt of this software and documentation. Please refer to that document for details on your rights and responsibilities in using this software.

Copyright © 1997-2003 Network Computing Technologies, Inc. All rights reserved.
Manufactured and produced solely in the USA.

Disclaimer

The information contained in this document is subject to change without notice. The statements and technical information contained herein are believed to be accurate and reliable, however Network Computing Technologies does not express or imply warranty. Users are responsible for the application of any products specified in this document.

Trademarks

Network Computing Technologies is a trademark. NCT, TrapGen, ev2t, procmon, the Trap Receiver and Network Computing Technologies logos, and “More Than Up” are trademarks of Network Computing Technologies, Inc. All other trademarks and registered trademarks are the property of their respective owners.

Network Computing Technologies, Inc.

*344 Maple Ave W, Suite 214
Vienna, VA 22180*

Web Site: www.nct-va.com

Table of Contents

ABOUT THIS GUIDE2

AUDIENCE.....2

SOFTWARE LICENSE.....2

DISCLAIMER.....2

TRADEMARKS2

LOGICAL VIEW REPORT5

LOGICAL VIEW5

Builder5

Pdu.....5

V1Pdu6

V1GetPdu6

V1TrapPdu6

V2Notifications6

V2Trap.....7

InformPdu.....7

GenericPacket7

Packet7

Subclass of Packet9

V1SetPdu9

Varbind.....9

IntVarbind.....9

StringVarbind9

IpAddrVarbind.....9

TimeTickVarbind9

ScalarVarbind.....9

CounterVarbind.....9

GaugeVarbind9

Scalar64Varbind.....9

OidVarbind10

Counter64Varbind.....10

OpaqueVarbind10

SnmpParser10

V1GetNextPdu10

V1ResponsePdu10

VarbindList.....10

VbPair.....11

Thread.....11

Poller12

Pdu.....12

V1TrapPdu13

V1GetPdu14

V1Pdu14

V2Notifications14

InformPdu.....14

V2Trap.....14

V1GetNextPdu15

V1ResponsePdu15

V1SetPdu15

Builder15

IntVarbind.....15

LOGICAL VIEW REPORT

StringVarbind 15
IpAddrVarbind 15
TimeTickVarbind 16
CounterVarbind 16
Scalar64Varbind 16
Counter64Varbind 16
ScalarVarbind 16
GaugeVarbind 16
OpaqueVarbind 17
Varbind 17
OidVarbind 17
TOTALS: 18

LOGICAL VIEW REPORT

Logical View

Builder

Public Operations:

virtual void Build ()

This is the abstract interface that many classes inherit and provides the BER underneath.
Takes an unsigned char pointer reference as a parameter.

Pdu

Abstract view of PDU part of an SNMP message

Derived from Builder

Public Operations:

int RequestId ()

Returns the value of the request ID of the SNMP message.

void RequestId(int) ()

Allows the setting of the Request ID field of the SNMP message.

int ErrorIndex ()

Returns the value of the error index field of the SNMP message.

void ErrorIndex(int) ()

Allows the setting of the Error Index field of the SNMP message.

int ErrorStatus ()

Returns the value of the error status field of the SNMP message.

void ErrorStatus(int) ()

Allows the setting of the Error Status field of the SNMP message.

int GenericType ()

Returns the value of the generic trap type field of the SNMP message (when it is a V1 trap type pdu).

void GenericType(int) ()

Allows the setting of the generic trap type field of the SNMP message (when it is a V1 trap type pdu).

int SpecificType ()

Returns the value of the specific trap type field of the SNMP message (when it is a V1 trap type pdu).

void SpecificType(int) ()

Allows the setting of the specific trap type field of the SNMP message (when it is a V1 trap type pdu).

unsigned long TimeTicks ()

Returns the value of the TimeTicks field of the SNMP message (when it is a V1 trap type pdu).

void TimeTicks(unsigned long) ()

Allows the setting of the timeticks field of the SNMP message (when it is a V1 trap type pdu).

const char* SenderIp ()

Returns the value of the senders IP address of the SNMP message (Sender's IP address field of V1 Trap, or socket level IP address of other types).

void SenderIP(const char*) ()

Allows the setting of the senders IP address field of the SNMP message (when it is a V1 trap type pdu).

const char* SenderOID ()
Returns the sender's OID field of a V1 trap.

void SenderOID(const char*) ()
Allows the setting of the sender's OID field of the SNMP message (when it is a V1 trap type pdu).

unsigned int Length ()
Pure virtual method to be supported by all pdu derived types. Returns the length in bytes of the pdu.

void Parse(unsigned char*&) ()
Pure virtual method to be supported by all pdu derived types. Calls the parse method of all member variables and stores the BER representation in the unsigned char* parameter.

unsigned char Type ()
Returns the pdu type. Also pure virtual as only the derived types know their type.

void opname4 ()

V1Pdu

This class represents a generic V1 Get PDU

[Derived from Pdu](#)

Private Attributes:

IntVarbind* mRequestIdVb

Holds the request id value and BER encoding of the SNMP message.

IntVarbind* mErrorStatusVb

Holds the error status value and BER encoding of the SNMP message.

IntVarbind* mErrorIndexVb

Holds the error index value and BER encoding of the SNMP message.

unsigned char mType

Holds the value of the pdu type.

Public Operations:

void MakeResponsePdu ()

Transform the SNMP message into an SNMP response.

V1GetPdu

The class is for generic V1 SNMP GET operations

[Derived from V1Pdu](#)

V1TrapPdu

This class is for SNMP V1 Trap messages

[Derived from Pdu](#)

V2Notifications

This class is for SNMPv2 Traps (Notifications)

[Derived from V1Pdu](#)

Private Attributes:

long mPeerAddr

To hold the IP address of the source of the notification.

Public Operations:

void SenderIp(struct sockaddr_in*) ()

Allows setting of the senders ip address from the lower layer socket data.

V2Trap

This class is used to make SNMPv2 Traps (notifications)

Derived from V2Notifications

InformPdu

This class handles SNMP INFORM Messages

Derived from V2Notifications

GenericPacket

Derived from Builder

Private Attributes:

int mSequenceNumber

Used for tracking is needed.

time_t mTimeStamp

used to track packets by time if needed.

unsigned long mPeerAddress

To indicate from where the packet was received if needed.

mutex mPacketLock

Used to lock the packet if needed.

GenericPacket* Next

Helper variable for linking packets together.

Public Operations:

time_t TimeStamp ()

Accessor for private member mTimeStamp.

void TimeStamp (time_t)

Mutator for private member mTimeStamp

unsigned long PeerAddress ()

Accessor for private member mPeerAddress

void PeerAddress (unsigned long)

Mutator for private member mPeerAddress

Packet

Derived from GenericPacket

Private Attributes:

StringVarbind* mCommunityVb
Holds the Community String in Varbind Form (TLV) which is needed for all SNMP (v1/v2c) packets.

IntVarbind* mVersionVb
Holds the version in Varbind Form (TLV) which is needed for all SNMP (v1/v2c) packets.

unsigned int mType
Holds the actual SNMP packet type. These types are defined in snmptypes.h.

Pdu* mPdu
Holds the appropriate Pdu which contains the remaining part of the SNMP message

Packet* Next
Helper function for linking Packets together. Seems this is redundant with the one in GenericPacket.

Public Operations:

return opname (argname)
const char* PktLibVersion ()
Returns the current version of the library.

void Type (unsigned int)
Allows you to set the packet type. Use with caution. It is better to use the constructor that takes the type as a parameter.

int Version ()
Returns the SNMP version.

void Version (int)
Sets the SNMP version for this packet.

void Version (Varbind*)
Sets the SNMP version for this packet.

unsigned int Length ()
Returns the Length of the Packet portion of the SNMP message.

unsigned int TotalLength ()
Returns the length of the total SNMP message.

const char* Community ()
Returns the community string of the SNMP message.

void Community (const char*)
Sets the Community string for the current packet.

void Community (StringVarbind*)
Sets the Community string of the current SNMP message.

int VbListLength ()
Returns the number of VbPairs on the list.

const char* VbOID (int)
Returns a string representation of the OID for the varbind specified as the parameters (1 based).

const char* VbType (int)
Returns a string representation of the Data Varbind Type (OID Varbind in a VbPair is always OID) for the varbind specified as the parameters (1 based).

const char* VbData (int)
Returns a string representation of the data for the varbind specified as the parameter (1 based).

VarbindList* VbList ()
Returns the VarbindList for this SNMP message.

void VbList (VarbindList*)
Sets the Varbind List for the SNMP message.

Pdu* pdu ()
Returns the Pdu that is part of this SNMP message.

void pdu (Pdu*)
Sets the Pdu of the current SNMP message. This function is not really necessary.

void Add (VbPair*)

Adds a VbPair to the list of the SNMP message.

void AddV2TrapVarbinds (unsigned long, char*, int, int)

This maps V1 parameters like specific type and generic type to V2 varbinds, and inserts them at the front of the list.

Private Operations:

VbPair* GetVbNumber (int)

Returns the VbPair from the Varbind list. The numbering is 1 based.

Subclass of Packet

Derived from Packet

V1SetPdu

Derived from V1Pdu

Varbind

IntVarbind

Derived from ScalarVarbind

StringVarbind

Derived from Varbind

IpAddrVarbind

Derived from Varbind

TimeTickVarbind

Derived from ScalarVarbind

ScalarVarbind

Derived from Varbind

CounterVarbind

Derived from ScalarVarbind

GaugeVarbind

Derived from ScalarVarbind

Scalar64Varbind

Derived from Varbind

OidVarbind

Derived from Varbind

Counter64Varbind

Derived from Scalar64Varbind

OpaqueVarbind

Derived from Varbind

SnmpParser

Private Attributes:

BOOL mIown

Used to determine whether the generated packet (Class Packet*) is owned (so as to determine whether it should be freed).

Packet* mPacket

Pointer to the packet that is generated from a UDP message read off the wire.

unsigned int mIndex

Persistent variable used when accessing the network data buffer.

Public Operations:

Packet* packet ()

Function used to retrieve the parsed SNMP message. Returns NULL if network data was malformed or incomplete. Sets mIown to FALSE.

Private Operations:

void decodeLength(unsigned char*, long*) ()

Helper function to decode the length according to the BER.

void encodeLength(unsigned char*, long*) ()

Helper function to encode the length according to the BER.

void decodeTag(unsigned char*, long*) ()

Helper function to decode the tag according to the BER.

void encodeTag(unsigned char*, long*) ()

Helper function to encode the tag according to the BER.

VIGetNextPdu

Derived from V1Pdu

VIResponsePdu

Derived from V1Pdu

VarbindList

Derived from Builder

Private Attributes:

VbPair* mHead

Internal head of list.

Public Operations:

unsigned int Length ()
Returns the length in bytes for the contained OidVarbind and Varbind. This value is used during the BER encoding process.

void Add (VbPair*)
Adds a VbPair to the list.

void AddFirst (VbPair*)
Adds a VbPair to the front of the list (used for adding V2Trap specific varbinds).

VbPair* FirstVbPair ()
Returns the head of the list as a VbPair pointer. Use VbPair->Next attribute of the returned VbPair to get the next (and so on).

VbPair

Derived from Builder

Private Attributes:

Class OidVarbind mOidVarbind*
Class Varbind mVarbind*
VbPair* Next
Used for linking the VbPairs together in a list. This is handled in VarbindList.Add(VbPair*).

Public Operations:

unsigned int Length ()
Returns the length in bytes for the contained OidVarbind and Varbind. This value is used during the BER encoding process.

OidVarbind* OidVarbind ()
Returns the OID half of the varbind pair as a OidVarbind pointer.

void OidVarbind (OidVarbind*)
This sets the OID half of the varbind pair. Requires a pointer to an OidVarbind as the parameter.

Varbind* VarBind ()
Returns the varbind (non-OID) half of the varbind pair as a Varbind pointer.

void VarBind (Varbind*)
This sets the other (non-OID) half of the varbind pair. Requires a pointer to an Varbind as the parameter.

Thread

Public Operations:

void Start ()
Cause the Thread to begin execution.

void Stop ()
Causes the Thread to stop execution. Achieved via the Thread's state attribute.

void Resume ()
Causes the Thread to continue operation. Also achieved through the Thread's state attribute.

void Exit ()
Causes the thread to cease operation completely. This thread cannot resume.

Thread::State State ()
Returns the current state of the Thread.

void main ()
Pure virtual member that is the main loop of execution.

Poller

[Derived from Thread](#)

[Public Operations:](#)

const char* Host ()
Returns the host from which this poller is configured to retrieve data.

void Host ()
Sets the host from which this poller is to retrieve data.

int Port ()
Returns the port on the host from which this poller is configured to retrieve data (typically 161).

void Port ()
Sets the port from which this poller will retrieve data.

const char* Community ()
Returns the community string this poller is configured to use to retrieve data.

void Community ()
Sets the community string this poller will use to retrieve data.

int NumberOfOids ()
Returns the current number of OIDs this poller is configured to retrieve. Max is 32.

void AddOid ()
Configures the OIDs on which this poller will issue SNMP Get commands. Max is 32 OIDs.

void Interval ()
Sets the duration in seconds between polls.

void SetCallback ()
Sets the callback function that will be called when a response is received. Use for "C" style callbacks. Prototype is void (*PollerCallback)(Packet *);

void ProcessResponse ()
Use this member function if you derive from Poller to process responses (rather than using SetCallback, or in addition to using SetCallback).

Pdu

Abstract view of PDU part of an SNMP message

[Derived from Builder](#)

[Public Operations:](#)

int RequestId ()
Returns the value of the request ID of the SNMP message.

void RequestId(int) ()
Allows the setting of the Request ID field of the SNMP message.

int ErrorIndex ()
Returns the value of the error index field of the SNMP message.

void ErrorIndex(int) ()
Allows the setting of the Error Index field of the SNMP message.

int ErrorStatus ()
Returns the value of the error status field of the SNMP message.

void ErrorStatus(int) ()
Allows the setting of the Error Status field of the SNMP message.

int GenericType ()
Returns the value of the generic trap type field of the SNMP message (when it is a V1 trap type pdu).

void GenericType(int) ()
Allows the setting of the generic trap type field of the SNMP message (when it is a V1 trap type pdu).

int SpecificType ()
Returns the value of the specific trap type field of the SNMP message (when it is a V1 trap type pdu).

void SpecificType(int) ()
Allows the setting of the specific trap type field of the SNMP message (when it is a V1 trap type pdu).

unsigned long TimeTicks ()
Returns the value of the TimeTicks field of the SNMP message (when it is a V1 trap type pdu).

void TimeTicks(unsigned long) ()
Allows the setting of the timeticks field of the SNMP message (when it is a V1 trap type pdu).

const char* SenderIp ()
Returns the value of the senders IP address of the SNMP message (Sender's IP address field of V1 Trap, or socket level IP address of other types).

void SenderIP(const char*) ()
Allows the setting of the senders IP address field of the SNMP message (when it is a V1 trap type pdu).

const char* SenderOID ()
Returns the sender's OID field of a V1 trap.

void SenderOID(const char*) ()
Allows the setting of the sender's OID field of the SNMP message (when it is a V1 trap type pdu).

unsigned int Length ()
Pure virtual method to be supported by all pdu derived types. Returns the length in bytes of the pdu.

void Parse(unsigned char*&) ()
Pure virtual method to be supported by all pdu derived types. Calls the parse method of all member variables and stores the BER representation in the unsigned char* parameter.

unsigned char Type ()
Returns the pdu type. Also pure virtual as only the derived types know their type.

void opname4 ()

V1TrapPdu

This class is for SNMP V1 Trap messages

[Derived from Pdu](#)

[Private Attributes:](#)

IntVarbind* mGenericTypeVb
Holds the value and BER representation of the generic trap type.

IntVarbind* mSpecificTypeVb
Holds the value and BER representation of the specific trap type.

TimeTickVarbind* mTimeStampVb
Holds the value and BER representation of the timestamp field of the SNMP message.

OidVarbind* mOidVb
Holds the value and BER representation of the sender's OID.

IpAddrVarbind* mIpAddrVb
Holds the value and BER representation of the sender's IP address.

V1GetPdu

The class is for generic V1 SNMP GET operations

[Derived from V1Pdu](#)

V1Pdu

This class represents a generic V1 Get PDU

[Derived from Pdu](#)

Private Attributes:

IntVarbind* mRequestIdVb

Holds the request id value and BER encoding of the SNMP message.

IntVarbind* mErrorStatusVb

Holds the error status value and BER encoding of the SNMP message.

IntVarbind* mErrorIndexVb

Holds the error index value and BER encoding of the SNMP message.

unsigned char mType

Holds the value of the pdu type.

Public Operations:

void MakeResponsePdu ()

Transform the SNMP message into an SNMP response.

V2Notifications

This class is for SNMPv2 Traps (Notifications)

[Derived from V1Pdu](#)

Private Attributes:

long mPeerAddr

To hold the IP address of the source of the notification.

Public Operations:

void SenderIp (struct sockaddr_in*)

Allows setting of the senders ip address from the lower layer socket data.

InformPdu

This class handles SNMP INFORM Messages

[Derived from V2Notifications](#)

V2Trap

This class is used to make SNMPv2 Traps (notifications)

[Derived from V2Notifications](#)

V1GetNextPdu

Derived from V1Pdu

V1ResponsePdu

Derived from V1Pdu

V1SetPdu

Derived from V1Pdu

Builder

Public Operations:

virtual void Build ()

This is the abstract interface that many classes inherit and provides the BER underneath.
Takes an unsigned char pointer reference as a parameter.

IntVarbind

Derived from ScalarVarbind

StringVarbind

Derived from Varbind

Private Attributes:

unsigned char* mData

Holds the BER representation of the varbind's data.

Public Operations:

const char* Value ()

Accessor method to retrieve the varbinds BER data.

void Value (char*)

Mutator method to set the varbinds BER data.

IpAddrVarbind

Derived from Varbind

Private Attributes:

unsigned char* mData

Holds the BER representation of the varbind's data.

Public Operations:

const char* Value ()

Accessor method to retrieve the varbinds BER data.

void Value (char*)

Mutator method to set the varbinds BER data.

TimeTickVarbind

Derived from ScalarVarbind

CounterVarbind

Derived from ScalarVarbind

Scalar64Varbind

Derived from Varbind

Protected Attributes:

__int64 mValue

used to hold the computer representation of the value (not BER).

Public Operations:

void Value (__int64)

virtual member to set the objects internal computer (non-BER) value.

__int64 Value ()

virtual to return computer representation of data (not BER).

Counter64Varbind

Derived from Scalar64Varbind

ScalarVarbind

Derived from Varbind

Protected Attributes:

unsigned long mValue

used to hold the computer representation of the value (not BER).

Public Operations:

unsigned long Value (unsigned long)

virtual to return computer representation of data (not BER).

void Value (unsigned long)

virtual member to set the objects internal computer (non-BER) value.

GaugeVarbind

Derived from ScalarVarbind

OpaqueVarbind

Derived from Varbind

Varbind

Protected Attributes:

char* mPrintable

The data of the varbind in a printable form.

Private Attributes:

unsigned short mLength

Holds the length of the varbind.

unsigned short mType

The varbind type defined in snmptypes.h

const char* mTypeString

Holds the human readable representation of the varbind type.

Varbind* Next

Helper member variable to link varbinds together.

Public Operations:

unsigned short Type ()

Accessor method for the varbind's type as defined in snmptypes.h

unsigned short DataLength ()

Accessor method for the varbind's length member.

void DataLength(unsigned short) ()

Mutator method for the varbind's length member.

unsigned char* Data ()

Pure virtual accessor method to retrieve the varbind's internal BER representation. Used during the build process.

const char* Printable ()

Pure virtual that returns the human readable form of the varbind's value.

const char* TypeString ()

Accessor for the varbind's type in human readable form.

OidVarbind

Derived from Varbind

Private Attributes:

unsigned char* mData

Holds the BER representation of the varbind's data.

Public Operations:

const char* Value ()

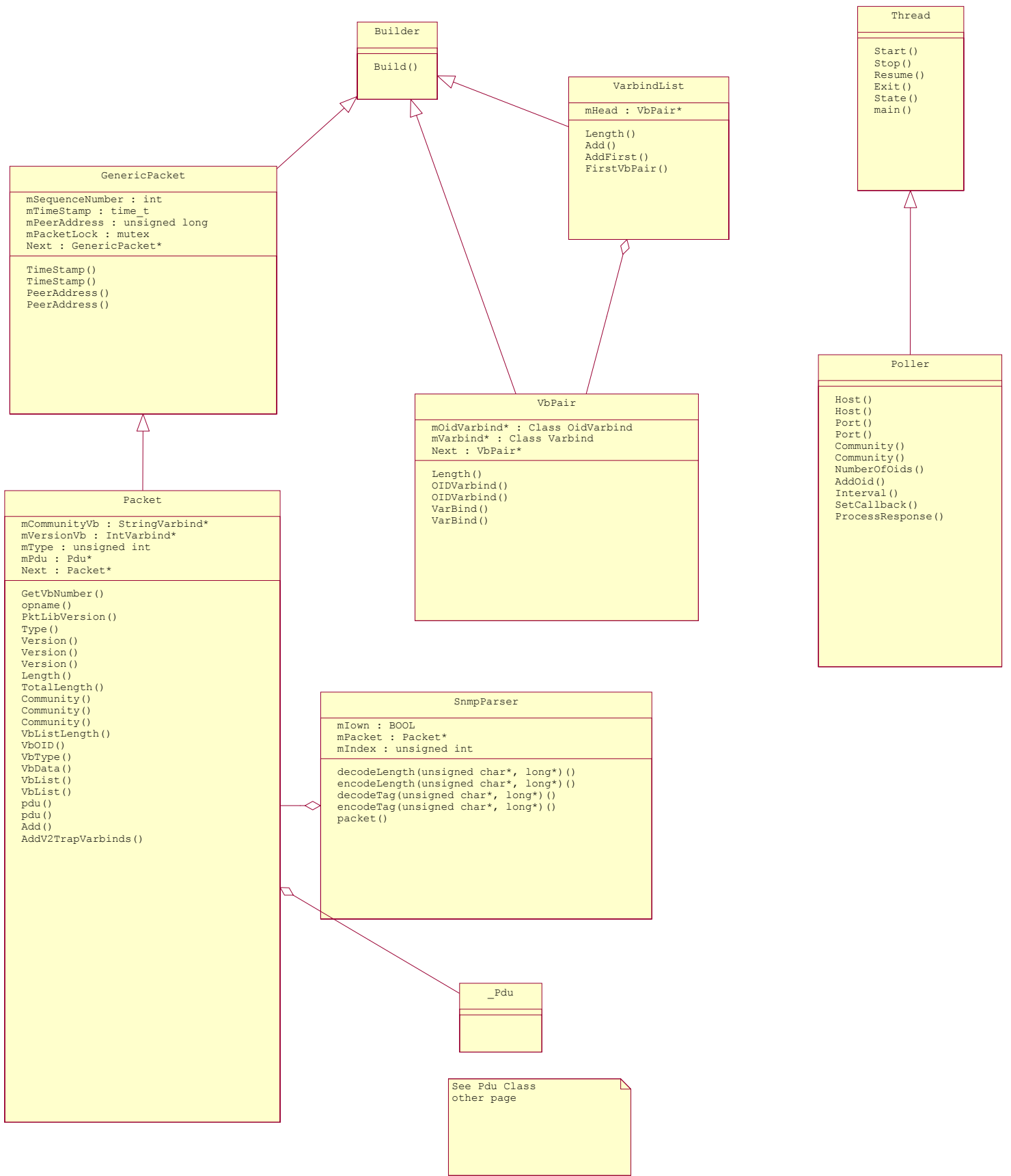
Accessor method to retrieve the varbinds BER data.

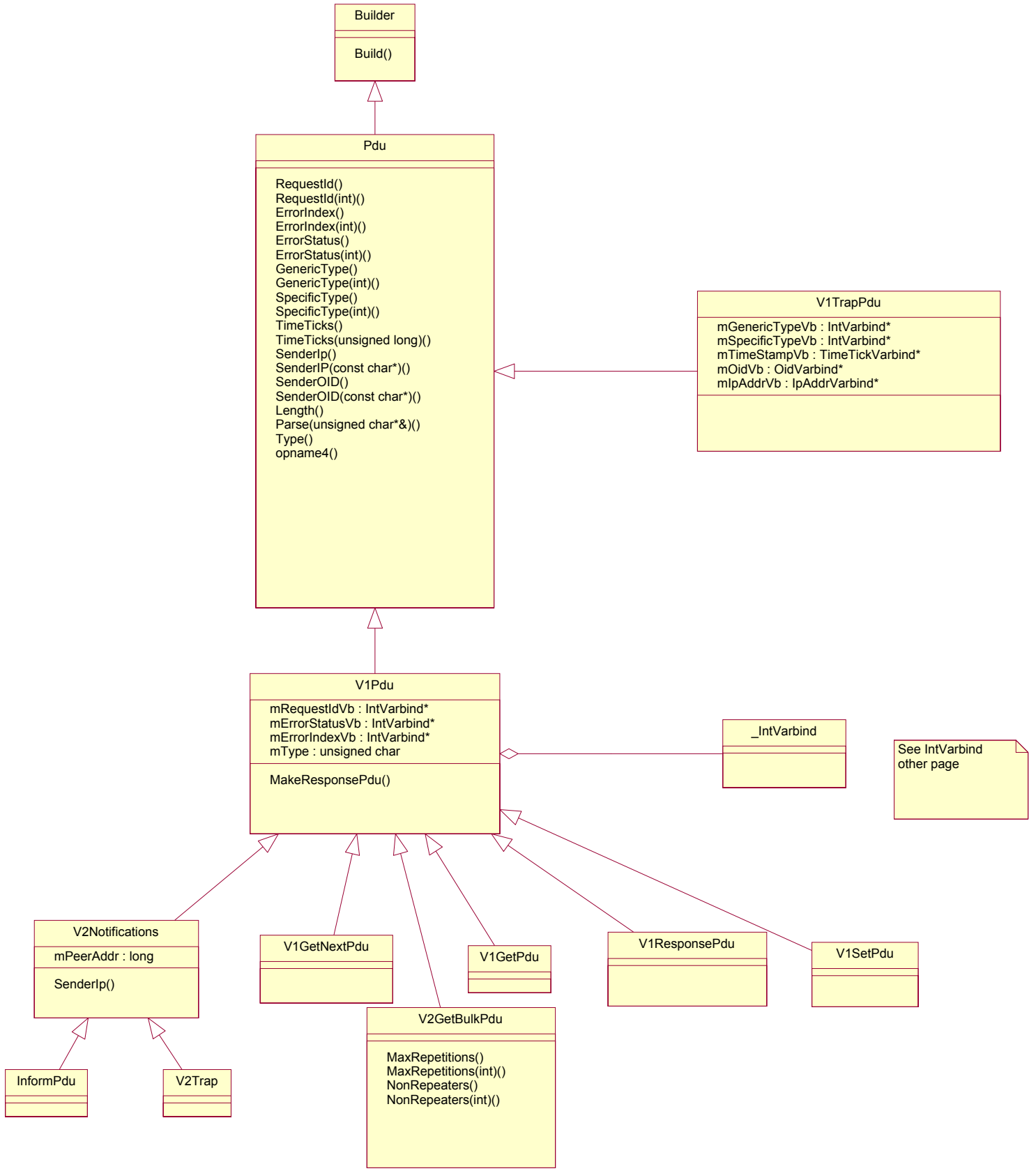
void Value (char*)

Mutator method to set the varbinds BER data.

TOTALS:

7 Logical Packages
54 Classes





See IntVarbind
other page

