HITACHI
DATA SYSTEMS

Hitachi HiCommand® Device Manager Virtual Disk Service Provider User's Guide

MK-95HC103-05

Trademarks

Notice of Export Controls

Export of technical data contained in this document may require an export license from the United States government and/or the government of Japan. Please contact the Hitachi Data Systems Legal Department for any export compliance questions.

Document Revision Level

Revision	Date	Description
MK-95HC103-00	August 2005	Initial Release
MK-95HC103-01	November 2005	Supersedes and replaces MK-95HC103-00
MK-95HC103-02	March 2006	Supersedes and replaces MK-95HC103-01
MK-95HC103-03	June 2006	Supersedes and replaces MK-95HC103-02
MK-95HC103-04	August 2006	Supersedes and replaces MK-95HC103-03
MK-95HC103-05	November 2006	Supersedes and replaces MK-95HC103-04

Changes in this Revision

- The Installation and operation section have been moved to Device Manager Server Installation manual.

Source Document(s) for this Revision

- HiCommand Device Manager VDS Provider User's Guide (3020-3-J10).

Preface

This manual describes Interface with Device Manager VDS Provider provided by HiCommand® Device Manager. In this manual, HiCommand Device Manager is abbreviated as Device Manager.

This manual is intended for developers using of Device Manager VDS Provider, a software product that provides information about storage subsystems (magnetic disk array devices) managed by Device Manager, for Virtual Disk Service (VDS), which runs on Windows Server™ 2003. Such users should have a basic knowledge of:

- VDS
- CIM (common information model)
- SANs (storage area networks)

Note: The use of Hitachi HiCommand Device Manager and all other Hitachi Data Systems products is governed by the terms of your agreement(s) with Hitachi Data Systems.

Software Revision Level

This document revision applies to HiCommand® Device Manager version 5.0 and higher.

Conventions for Storage Capacity Values

This manual uses the following convention for storage capacity values:

- 1 KB (kilobyte) is 1,024 bytes
- 1 MB (megabyte) is $1,024^2$ bytes
- 1 GB (gigabyte) is $1,024^3$ bytes
- 1 TB (terabyte) is $1,024^4$ bytes

Readme and Release Notes Contents

Readme and release notes files may be included on the installation CD. These files contain requirements and notes for use of HiCommand® Device Manager VDS Provider that may not be fully described in this manual. Review these files before installing HiCommand® Device Manager VDS Provider.

Referenced Documents

- *HiCommand Device Manager Web Client User's Guide,* MK-91HC001

Comments

Please send us your comments on this document. Make sure to include the document title, number, and revision. Please refer to specific section(s) and paragraph(s) whenever possible.

- **E-mail:** doc.comments@hds.com
- **Fax:** 858-695-1186
- **Mail:**
  Technical Writing, M/S 35-10
  Hitachi Data Systems
  10277 Scripps Ranch Blvd.
  San Diego, CA 92131

Thank you! (All comments become the property of Hitachi Data Systems Corporation.)

Contents

List of Tables

Chapter 1 Overview

This chapter provides an overview of Device Manager Virtual Disk Service (VDS) Provider.

- Overview of Device Manager VDS Provider (see section 1.1)
- Functions of Device Manager VDS Provider (see section 1.2)

1.1 Overview of Device Manager VDS Provider

Windows Server™ 2003 provides Virtual Disk Service, a virtual disk system designed to manage storage subsystem devices via standardized interfaces. Device Manager VDS Provider is a Device Manager software product that provides and configures storage subsystem information for VDS.

Device Manager VDS Provider supports the following operating systems:

- Windows Server 2003 (with Service Pack 1): For Enterprise Edition or Datacenter Edition, the 64-bit processor version (running on WOW64) is supported.
- Windows Server 2003 x64 Edition: EM64T (running on WOW64) is supported.

1.2 Functions of Device Manager VDS Provider

Device Manager VDS Provider is a hardware provider resident on a Windows host that replies to VDS requests. Device Manager VDS Provider provides functions for:

- Obtaining the following storage subsystem-related configuration information:
  - Subsystem
  - Logical unit number (LUN)
  - LUN replication information
  - Path information
- Performing the following storage subsystem-related configuration tasks:
  - LUN creation
  - LUN expansion
  - LUN deletion
  - LUN mapping
  - LUN masking

Before you can use the `DISKRAID` command provided by Microsoft, you must perform the following installation tasks:

- Install Device Manager VDS Provider
- Download VDS 1.1 SDK from the Microsoft web page (`http://www.microsoft.com/`) and then install it

Important: Readme and release notes files may be included on the installation CD. These files contain requirements and notes for use of HiCommand® Device Manager VDS Provider that may not be fully described in this manual. Review these files before installing HiCommand® Device Manager VDS Provider.

Important: To use Device Manager VDS Provider, you need to change the value of the Device Manager server property `server.cim.support` to `true` in the `server.properties` file, and then restart the Device Manager server.

Chapter 2 Interface

2.1 Interface with the HiCommand Device Manager server

VDS provider acquires and configures configuration information of subsystems using the HiCommand Device Manager server (hereinafter called HDvM).

This section describes the interface used to communicate with the HDvM server.

(1) CIM interface

The VDS provider operates subsystems via the CIM interfaceprovided by the HDvM server. While the CIM interface denotes the management targets as classes or instances, functions provided by them may be updated as new revisions of HDvM are released. However, in order to maintain upwards compatibility, functions provided by each revision are denoted as NAMESPACEs. Therefore, a function, which has been provided since a certain revision, can be used by specifying the appropriate NAMESPACE when using the CIM interface. The following table shows NAMESPACEs used for each revision by the HDvM VDS provider.

Table 2-1 The revision of HDvM and NAMESPACE

revision	namespace
02-40	root/hitachi/dm24
03-00	root/hitachi/dm30
03-01	root/hitachi/dm30
03-50	root/hitachi/dm35
04-00	root/hitachi/dm40
04-10	root/hitachi/dm41
04-20	root/hitachi/dm42
04-30	root/hitachi/dm43
05-00	root/hitachi/dm50
05-10	root/hitachi/dm51
05-50	root/hitachi/dm51

(2) Configuration file

The VDS provider communicates with the HDvM server using the CIM interface. There is a property called "server.cim.support" in the configuration file (server.properties) managed by the HDvM server. This property "server.cim.support" is set to "false" as the default. Change the setting value in the property "server.cim.support" to "true" before using the VDS provider, and then restart the HDvM server service to apply the change in the configuration file (server.properties).

2.2 Interface details

This section describes interfaces for the VDS client and VDS service provided by the HDvM VDS provider. Ones with "VDS_E_NOT_SUPPORTED" in the "Return value" column indicate the methods that are not supported by this version.

[Common return values among processing]

(1) Normal termination: -->S_OK

(2) insufficient memory: -->E_OUTOFMEMORY

(3) The method cannot be executed because another processing is being executed: -->VDS_E_ANOTHER_CALL_IN_PROGRESS

(4) The object no longer exists (except for the provider): -->VDS_E_OBJECT_DELETED

(5) The parameter value is out of range (unused parameters are also checked for their value ranges): -->E_INVALIDARG

(6) Not supported by Hitachi proprietary implementation: -->VDS_E_NOT_SUPPORTED

2.2.1 IVdsProvider

2.2.1.1 Summary of interface

Table 2-2 Summary of IVdsProvider

description	This interface supports the common function for hardware provider and softwareprovider. It can be used by VDS client with VDS service.
implementation	CHdvmProvider

2.2.1.2 Method specification

GetProperties

Table 2-3 Method specification of IVdsProvider::GetProperties

definition	HRESULT GetProperties(VDS_PROVIDER_PROV **pProviderProp*)
description	This returns the property of provider. It must be initialized by IVdsProviderPrivate::OnLoad.
parameter	[OUT] pProviderProp The buffer for receiving the property. It must not be NULL.
return value	Normal termination: S_OK (0) Abnormal termination: Status code for an error
implementation	This returns the structure of VDS_PROVIDER_PROV that has a

	fixed value. cf: 2.3.1
error	If pProviderProp is NULL: E_POINTER (80004003 hex) If it is not initialized by OnLoad: E_UNEXPECTED (8000FFFF HEX)

2.2.2 IVdsProviderPrivate

2.2.2.1 Summary of interface

Table 2.4 Summary of IVdsProviderPrivate

description	This supports the common function for hardware provider and software provider. It can be used by only VDS service
implementation	CHdvmProvider

2.2.2.2 Specification of method

GetObject

Table 2-5 Specification of IVdsProviderPrivate::GetObject

definition	HRESULT GetObject(VDS_OBJECT_ID *ObjectId*, VDS_OBJECT_TYPE *type*, IUnknown **ppObjectUnk)
description	This returns the specified VDS object. It must be initialized by OnLoad.
parameter	[IN] ObjectID GUID of the VDS object that will be got. [IN] type Type of the VDS object that will be got. [OUT] ppObjectUnk Pointer of the buffer for receiving the VDS object. It must not be NULL.
return value	Normal termination: S_OK (0) Abnormal termination: Status code for an error
implementation	It returns the VDS object that has the specified GUID and type in the buffer.
error	If ppObjectUnk is NULL: E_POINTER (80004003 hex) If the type does not exist: E_INVALIDARG (80070057 hex) If the GUID does not exist: VDS_E_OBJECT_NOT_FOUND (80042405 hex)

	If the specified GUID exists but the type does not exist: VDS_E_OBJECT_NOT_FOUND (80042405 hex) If it is not initialized by OnLoad: E_UNEXPECTED (8000FFFF HEX)

OnLoad

Table 2-6 Specification of IVdsProviderPrivate::OnLoad

definition	HRESULT OnLoad(LPWSTR pwszMachineName, IUnknown *pCallbackObject)
description	This is for initialization.
parameter	[IN] pwszMachineName Name of the machine that VDS service is running on. It must not be NULL. [IN] pCallbackObject The callback object for getting interfaces. It must not be NULL.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	The thread that initializes CHdvmSubSystem with the structure of HDVM_SUB_SYSTEM_PROP that has been got when the service started.
error	If pwszMachineName or pCallbackObject is NULL: E_POINTER (80004003 hex) If initialization is failed: E_FAIL(80004005 hex)

OnUnload

Table 2-7 Specification of IVdsProviderPrivate::OnUnload

definition	HRESULT OnUnload(BOOL bForceUnload)
description	It unloads the VDS provider on termination.
parameter	[IN] bForceUnload The flag for termination of provider and unloading of it forcefully.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	It always returns S_OK.
error	No error

2.2.3 IVdsHwProvider

2.2.3.1 Summary of interface

Table 2-8 Summary of IVdsHwProvider

description	It supports the particular operation for hardware provider. It can be used by VDS client.
implementation	CHdvmProvider

2.2.3.2 Specification of method

QuerySubSystems

Table 2-9 Specification of IVdsHwProvider::QuerySubSystems

definition	HRESULT QuerySubSystems(IEnumVdsObject ***ppEnum*)
description	It returns the list of subsystems that the provider manages. It must be initialized by IVdsProviderPrivate::OnLoad.
parameter	[OUT] ppEnum The pointer of the buffer for receiving the objects of the subsystems that the provider has. It must not be NULL.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	If the operation of receiving the information of subsystems is completed on requirement: It returns the list of subsystems in the buffer. If the operation of receiving the information of subsystems is not completed on requirement It returns the list of objects that have no items.
error	If ppEnum is NULL: E_POINTER (80004003 hex) If it is not initialized by OnLoad: E_UNEXPECTED (8000FFFF HEX)

Reenumerate

Table 2-10 Specification of IVdsHwProvider::Reenumerate

definition	HRESULT Reenumerate()
description	It discovers the subsystems that is connected or disconnected.
parameter	No parameters
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	It cleans the internal buffer and reads all of the information of subsystems again.
error	If the operation of getting all of the data of subsystems is processing: E_FAIL (80004005 hex)

Refresh

Table 2-11 Specification of IVdsHwProvider::Refresh

definition	HRESULT Refresh()
description	It loads the data of subsystems again. It must be initialized by IVdsProviderPrivate::OnLoad.
parameter	No parameters
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	It cleans the internal buffer and reloads all of the data of subsystems.
error	If the operation for getting all of the data of subsystems are processing: E_FAIL (80004005 hex) If it is not initialized by OnLoad: E_UNEXPECTED (8000FFFF HEX)

2.2.4 IVdsHwProviderPrivate

2.2.4.1 Summary of interface

Table 2-12 Specification of IVdsHwProviderPrivate

description	It supports the particular operation of hardware provider. It can be used by only VDS service.
implementation	CHdvmProvider

2.2.4.2 Specification of method

QueryIfCreatedLun

Table 2-13 IVdsHwProviderPrivate::QueryIfCreatedLun

definition	HRESULT QueryIfCreatedLun(LPWSTR pwszDevicePath, VDS_LUN_INFORMATION **pVdsLunInformation*, VDS_OBJECT_ID **pLunId*)
description	It requires whether the specified LUN exits. It must be initialized by IVdsProviderPrivate::OnLoad.
parameter	[IN] pwszDevicePath The path for the device. [IN] pVdsLunInformation The information of device. It must not be NULL. [OUT] pLunId If the provider has the LUN, this returns GUID. If not, this returns GUID_NULL.
return value	Normal termination: If the provider has it: S_OK (0), If not: S_FALSE (1) Abnormal termination: status code of error
implementation	If the LUN is stored in the internal buffer, this returns the GUID. If not, this returns GUID_NULL.
error	If pVdsLunInformation is NULL: E_POINTER (80004003 HEX) If this is not initialized by OnLoad: E_UNEXPECTED (8000FFFF HEX)

2.2.5 IVdsHwProviderType

2.2.5.1 Summary of interface

Table 2-14 IVdsHwProviderType

description	This supports the operation for getting the kind of hardware provider. It can be used with VDS service by VDS client.
implementation	CHdvmProvider

2.2.5.2 Specification of method

GetProviderType

Table 2-15 IVdsHwProviderType::GetProviderType

definition	HRESULT GetProviderType(VDS_HWPROVIDER_TYPE *pType)
description	This acquires the kind of hardware provider. It must be initialized by IVdsProviderPrivate::OnLoad.
parameter	[OUT] pType The pointer of the buffer that receives the kind of hardware provider. It must not be NULL.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This always returns VDS_HWT_FIBRE_CHANNEL.
error	If pType is NULL: E_POINTER (80004003 HEX) If it is not initialized by OnLoad: E_UNEXPECTED (8000FFFF HEX)

2.2.6 IVdsSubSystem

2.2.6.1 Summary of interface

Table 2-16 IVds HwProviderType Interface Summary

description	This supports the operation for subsystems.
implementation	CHdvmSubSystem

2.2.6.2 Specification of method

GetProperties

Table 2-17 IVdsSubSystem::GetProperties

definition	HRESULT GetProperties(VDS_SUB_SYSTEM_PROP *pSubSystemProp)
description	This returns the property of subsystem.
parameter	[OUT] pSubSystemProp The buffer for receiving the property of subsystems. It must not be NULL.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This returns the data of the structure VDS_SUB_SYSTEM_PROP. cf.2.3.2
error	If pSubSystemProp is NULL: E_POINTER (80004003 hex)

GetProvider

Table 2-18 IVdsSubSystem::GetProvider

definition	HRESULT GetProvider(IVdsProvider ***ppProvider)*
description	This returns the provider that has the subsystem.
parameter	[OUT] ppProvider The pointer of the buffer for storing the object of the provider that has the subsystem. It must not be NULL.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This returns the object of the provider that is stored in the internal buffer.
error	If ppProvider is NULL: E_POINTER (80004003 hex)

QueryControllers

Table 2-19 IVdsSubSystem::QueryControllers

definition	HRESULT QueryControllers(IEnumVdsObject ***ppEnum*)
description	This returns the list of the controllers of subsystem. If no controllers exist, this returns the empty of Enumeration.
parameter	[OUT] ppEnum The pointer of the buffer for storing the enumeration object of controllers that the subsystem has. It must not be NULL.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This returns the list of controllers that are stored in the internal buffer.
error	If ppEnum is NULL: E_POINTER (80004003 hex)

QueryLuns

Table 2-20 IVdsSubSystem::QueryLuns

definition	HRESULT QueryLuns(IEnumVdsObject ***ppEnum*)
description	This returns the list of LUNs of subsystems. If no LUNs exist, this returns the empty of Enumeration.
parameter	[OUT] ppEnum The pointer of buffer for storing the enumeration objects that the subsystem has. It must not be NULL.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error

implementation	This returns the list of LUN objects that are stored in the internal buffer.
error	If ppProvider is NULL: E_POINTER (80004003 hex)

QueryDrives

Table 2-21 IVdsSubSystem::QueryDrives

definition	HRESULT QueryDrives(IEnumVdsObject ***ppEnum*)
description	This returns the list of the drives of subsystem. If no drives exist, this returns the empty of Enumeration.
parameter	[OUT] ppEnum The pointer of buffer for storing the enumeration objects that the subsystem has. It must not be NULL.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This returns the list of the drives that are stored in the internal buffer.
error	If ppEnum is NULL: E_POINTER (80004003 hex)

GetDrive

Table 2-22 IVdsSubSystem::GetDrive

definition	HRESULT GetDrive(SHORT sInternalBusNumber, SHORT sSlotNumber, IVdsDrive ***ppDrive*)
description	This returns the specified drive.
parameter	[IN] sInternalBusNumber The number of bus for the target drive. [IN] sSlotNumber The number of slot for the target drive. [OUT] ppDrive The pointer of buffer for storing the objects of drives.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This returns the drive objects that is valid for the specified the number of bus and the number of slot.
error	If both pwszUnmaskingList and ppAsyncare are NULL: E_POINTER (80004003 HEX) If the specified drive does not exist: VDS_E_OBJECT_NOT_FOUND (80042405 HEX)

Reenumerate

Table 2-23 IVdsSubSystem::Reenumerate

definition	HRESULT Reenumerate()
description	This loads the information of subsystems that HDvM manages at the time as the information that the provider manages.
parameter	No parameters
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This cleans the internal buffer of the subsystem and reloads the information of it.
error	If the information of the subsystem fails to be loaded: E_FAIL(80004005 HEX)

SetControllerStatus

Table 2-24 IVdsSubSystem::SetControllerStatus

definition	HRESULT SetControllerStatus(VDS_OBJECT_ID *pOnlineControllerIdArray, LONG lNumberOfOnlineControllers, VDS_OBJECT_ID *pOfflineControllerArray, LONG lNumberOfOfflineControllers)
description	This sets the online or offline for the controller of subsystem
parameter	[IN] pOnlineControllerIdArray The array of GUID for the online controller. If lNumberOfOnlineControllers is not 0, it must not be NULL. [IN] lNumberOfOnlineControllers The number of controllers that are included in pOnlineControllerIDArray. [IN] pOfflineControllerArray The array of GUID for the offline controllers. If lNumberOfOfflineControllers is not 0, it must not be NULL. [IN] lNumberOfOfflineControllers The number of controllers that are included in pOfflineControllerIDArray.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This always returns VDS_E_NOT_SUPPORTED..
error	VDS_E_NOT_SUPPORTED (80042400 HEX)

CreateLun

Table 2-25 IVdsSubSystem::CreateLun

definition	HRESULT CreateLun(VDS_LUN_TYPE type ULONGLONG ullSizeInBytes, VDS_OBJECT_ID *pDriveIdArray, LONG lNumberOfDrives, LPWSTR pwszUnmaskingList, VDS_HINTS *pHints, IVdsAsync **ppAsync)
description	This creates LUN.
parameter	[IN] type The type of the created LUN [IN] ullSizeInBytes The size of the created LUN. If the list of drives are included in pDriveIdArray, it may be 0. [IN] pDriveIdArray The array of GUID for the drives for creating LUN. If it is NULL, the provider must create LUN with the valid drives. The provider of the current version supports the creation of LUN with any drives and the parameter is used only for checking whether the specified drives exist. [IN] lNumberOfDrives The number of drives that are included in pDriveIdArray. If it is 0, the provider must create LUN with the valid drives. The provider of the current version supports the creation of LUN with any drives and the parameter is used only for checking whether the specified drives exist. [IN] pwszUnmaskingList This is the node for setting the authorization to the created LUN. This can store the names that are divided with ":". If the empty string "" is specified, the LUN cannot be accessed from any nodes. It must not be NULL. It is unavailable on the provider of the current version. [IN] pHints It is the hint of creation of LUN. The parameter is set as the lower priority than the others. It must not be NULL. It is unavailable on the provider of the current version. [OUT] ppAsync The pointer of buffer for storing the unsynchronized objects that the caller confirms the status of operation for. It must not be NULL.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	(1) Creating volume for the pool of the subsystem (2) Updating the information of subsystem.
error	If ppAsync is NULL:

	E_POINTER (80004003 HEX) If ullSizeInBytes is 0: E_INVALIDARG If the capacity for creating the specified size LUN is not sufficient: VDS_E_NOT_ENOUGH_SPACE (8004240F HEX) If LUN failed to be created: E_FAIL(80004005 HEX) If the drive that is specified in PDriveIdArray does not exist: VDS_E_NOT_SUPPORTED (80042400 HEX)

ReplaceDrive

Table 2-26 IVdsSubSystem::ReplaceDrive

definition	HRESULT ReplaceDrive(VDS_OBJECT_ID DriveToBeReplaced VDS_OBJECT_ID *ReplacementDrive*)
description	Replacing the drive of subsystem to the other drive or connecting it to the other.
parameter	[IN] DriveToBeReplaced GUID of the original drive [IN] ReplacementDrive GUID of the new drive
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This always returns VDS_E_NOT_SUPPORTED..
error	VDS_E_NOT_SUPPORTED (80042400 HEX)

SetStatus

Table 2-27 IVdsSubSystem::SetStatus

definition	HRESULT SetStatus(VDS_SUB_SYSTEM_STATUS *status*)
description	This sets the status of subsystem.
parameter	[IN] status The new status of subsystem
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This always returns VDS_E_NOT_SUPPORTED..
error	VDS_E_NOT_SUPPORTED (80042400 HEX)

QueryMaxLunCreateSize

Table 2-28 IVdsSubSystem::QueryMaxLunCreateSize

definition	HRESULT QueryMaxLunCreateSize(VDS_LUN_TYPE *type*, VDS_OBJECT_ID **pDriveIdArray*, LONG lNumberOfDrives, VDS_HINTS **pHints*, ULONGLONG *pullMaxLunSize)
description	This returns the maximum capacity of the LUN that can be created with the type and the hint.
parameter	[IN] type The type of the created LUN [IN] pDriveIdArray The array of GUID of the drive for creating LUN. If the parameter is NULL, the provider creates LUN by finding the valid drive automatically. The provider of the current version supports only the sum of the maximum capacity for creating LUN with any drives and the parameter is unavailable. [IN] lNumberOfDrives The number of the drives in pDriveIdArray. If pDriveIdArray is NULL, the parameter is set to 0. The provider of the current version supports only the sum of the maximum capacity for creating LUN with any drives and the parameter is unavailable. [IN] pHints Showing the hints of creating LUN. The parameter is the lower priority to the others. It must not be NULL. In the current provider, the parameter is unavailable. [OUT] pullMaxLunSize The buffer for receiving the maximum size to create LUN. It must not be NULL.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This returns the maximum size by getting HITACHI_GetSupportedSizes. If the size is smaller than HITACHI_GetSupportedSizeRange, this returns the got size as the maximum capacity. And if the size is larger, this returns GetSupportedSizeRange.
error	If pullMaxLunSize is NULL E_POINTER (80004003 HEX)

2.2.7 IVdsSubSystemNaming

2.2.7.1 Summary of interface

Table 2-29 IVdsSubSystemNaming

description	This supports the operation for setting the name of subsystem.
implementation	CHdvmSubSystem

2.2.7.2 Specification of method

SetFriendlyName

Table 2-30 IVdsSubSystemNaming::SetFriendlyName

definition	HRESULT SetFriendlyName(LPWSTR *pwszName*)
description	This sets the name of subsystem.
parameter	[IN] pwszName The name of subsystem. The maximum length that the end of string is included in is 64 charactors in Unicode. It must not be NULL.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This calls HITACHI_StorageSystem.SetProperty(ElementName). This does not check the length of string and it is based in the specification of CIM/WBEM.
error	If pwszName is NULL E_POINTER (80004003 HEX) If failed: E_FAIL(80004005 HEX)

2.2.8 IVdsController

2.2.8.1 Summary of interface

Table 2-31 IVdsController

description	This supports the operation of controller.
implementation	CHdvmController

2.2.8.2 Specification of method

GetProperties

Table 2-32 IVdsController::GetProperties

definition	HRESULT GetProperties(VDS_CONTROLLER_PROP **pControllerProp*)
description	This returns the property of controller.

parameter	[OUT] pControllerProp The buffer for receiving the property of controller. It must not be NULL.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This returns the structure VDS_CONTROLLER_PROP in the internal buffer. cf. 2.3.3
error	If pControllerProp is NULL: E_POINTER (80004003 hex)

GetSubSystem

Table 2-33 IVdsController::GetSubSystem

definition	HRESULT GetSubSystem(IVdsSubSystem ***ppSubSystem*)
description	This returns the subsystem that the controller belongs to.
parameter	[OUT] ppSubSystem The pointer of buffer for receiving the object of subsystem that the controller belongs to. It must not be NULL.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This returns the object of subsystem in the internal buffer.
error	If ppSubSystem is NULL, E_POINTER (80004003 hex)

FlushCache

Table 2-34 IVdsController::FlushCache

definition	HRESULT FlushCache()
description	This flushes the cache of controller to the subsystem.
parameter	No parameters.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This always returns VDS_E_NOT_SUPPORTED.
error	VDS_E_NOT_SUPPORTED (80042400 HEX)

InvalidateCache

Table 2-35 IVdsController::InvalidateCache

definition	HRESULT InvalidateCache()
description	This makes the cache of controller unavailable.
parameter	No parameters

return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This always returns E_NOTIMPL.
error	E_NOTIMPL (80004001 hex)

Reset

Table 2-36 IVdsController::Reset

definition	HRESULT Reset()
description	This makes the cache unavailable by re-initializing the controller.
parameter	No parameters.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This always returns E_NOTIMPL.
error	E_NOTIMPL (80004001 hex)

SetStatus

Table 2-37 IVdsController::SetStatus

definition	HRESULT SetStatus(VDS_CONTROLLER_STATUS *status*)
description	This sets the status of controller.
parameter	[IN] status The new status of controller.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This always returns VDS_E_NOT_SUPPORTED..
error	VDS_E_NOT_SUPPORTED (80042400 HEX)

2.2.9 IVdsControllerControllerPort

2.2.9.1 Summary of interface

Table 2-38 IVdsControllerControllerPort

description	This supports the operation of enumeration of controller port.
implementation	CHdvmController

2.2.9.2 Specification of method

QueryControllerPorts

Table 2-39 IVdsControllerControllerPort::QueryControllerPorts

definition	HRESULT QueryControllerPorts (IEnumVdsObject** *ppEnum*)
description	This returns the list of controller port.
parameter	[OUT] ppEnum The pointer of buffer for receiving the list of controller ports. It must not be NULL.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This returns the list of ports that are stored in the internal buffer of the controller.
error	If ppEnum is NULL: E_POINTER (80004003 HEX)

2.2.10 IVdsContollerPort

2.2.10.1 Summary of interface

Table 2-40 IVdsControllerPort

description	This supports the operation of port.
implementation	CHdvmPort

2.2.10.2 Specification of method

GetController

Table 2-41 IVdsControllerPort::GetController

definition	HRESULT GetController(IVdsController ***ppController*)
description	This returns the controller of port.
parameter	[OUT] ppController The pointer of buffer for receiving the object of controller that has the ports. It must not be NULL.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This returns the object of controller that is stored in the internal

	buffer.
error	If ppController is NULL: E_POINTER (80004003 HEX)

GetProperties

Table 2-42 IVdsControllerPort::GetProperties

definition	HRESULT GetProperties(VDS_PORT_PROP **pPortProp*)
description	This returns the property of port.
parameter	[OUT] pPortProp The buffer for receiving the property of port. It must not be NULL.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This returns the structure VDS_PORT_PROP that is stored in the internal buffer. Cf. 2.3.4.
error	If pPortProp is NULL: E_POINTER (80004003 HEX)

QueryAssociatedLuns

Table 2-43 IVdsControllerPort::QueryAssociatedLuns

definition	HRESULT QueryAssociatedLuns(IEnumVdsObject ***ppEnum*)
description	This returns the list of LUNs that are active and are assigned with the port.
parameter	[OUT] ppEnum The pointer of buffer for receiving the objects of enumeration of LUNs that are assigned with the port. It must not be NULL.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This returns the list of LUNs that are assigned with the port by finding the paths that are associated with the port in the internal buffer.
error	If ppEnum is NULL: E_POINTER (80004003 HEX)

Reset

Table 2-44 IVdsControllerPort::Reset

definition	HRESULT Reset()

description	This reinitialized the port.
parameter	No parameters.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This always returns VDS_E_NOT_SUPPORTED..
error	VDS_E_NOT_SUPPORTED (80042400 HEX)

SetStatus

Table 2-45 IVdsControllerPort::SetStatus

definition	HRESULT SetStatus(VDS_PORT_STATUS *status*)
description	This sets the status of port.
parameter	[IN] status The new status of port
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This always returns VDS_E_NOT_SUPPORTED..
error	VDS_E_NOT_SUPPORTED (80042400 HEX)

2.2.11 IVdsLun

2.2.11.1 summary of interface

Table 2-46 IVdsLun

description	This supports the operation of LUN. The LUN of VDS provider is StorageVolume of CIM provider.
implementation	CHdvmLun

2.2.11.2 specification of method

GetProperties

Table 2-47 IVdsLun::GetProperties

definition	HRESULT GetProperties(VDS_LUN_PROP **pLunPro*p)
description	This returns the property of LUN.
parameter	[OUT] pLunProp The buffer of LUN property. It must not be NULL.

return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This returns the structure VDS_LUN_PROP in the internal buffer. Cf.2.3.5.
error	If pLunProp is NULL: E_POINTER (80004003 hex)

GetSubSystem

Table 2-48 IVdsLun::GetSubSystem

definition	HRESULT GetSubSystem(IVdsSubSystem ***ppSubSystem*)
description	This returns the subsystem that LUN belongs to.
parameter	[OUT] ppSubSystem The pointer of buffer for receiving the object of subsystem that has the LUN. It must not be NULL.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This returns the object of subsystem in the internal buffer.
error	If ppSubSystem is NULL: E_POINTER (80004003 hex)

GetIdentificaitonData

Table 2-49 IVdsLun::GetIdentificationData

definition	HRESULT GetIdentificationData(VDS_LUN_INFORMATION **pLunInfo*)
description	This returns the distinction of LUN. It is consisted of SCSI Inquiry and VPD Page80h,83h.
parameter	[OUT] pLunInfo The buffer for LUN. It must not be NULL.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This returns the structure VDS_LUN_INFORMATION in the internal buffer. Cf:2.3.6.
error	If pLunInfo is NULL: E_POINTER (80004003 hex)

Extend

Table 2-50 IVdsLun::Extend

definition	HRESULT Extend(ULONGLONG ullNumberOfBytesToAdd,

	VDS_OBJECT_ID **pDriveIdArray*, LONG lNumberOfDrives, IVdsAsync ***ppAsync*)
description	This extends LUN. It is not supported only for RAID400.
parameter	[IN] ullNumberOfBytesToAdd The bytes of LUN to be extended [IN] pDriveIdArray The array of GUID of drives that LUNs are extended for. If NULL is specified, the provider must extend LUNs by finding the valid drives. In the current version, the provider supports the LUN extension only with any drives. The parameter is unavailable. [IN] lNumberOfDrives The number of drives in pDriveIdArray. If pDriveIdArray is NULL, 0 should be specified as the parameter. The current provider supports the LUN extension with any drives and the parameter is unavailable. [OUT] ppAsync The pointer of buffer for the unsynchronized object. It must not be NULL.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	(1) Execute HITACHI_StorageConfigurationService.CreateOrModifyElementFromPool(). As the parameter "TheElement", HITACHI_StorageVolume that is equal to the LUN is used. (2) Generates IVdsAsync and returns it.
error	If ppAsync is NULL: E_POINTER (80004003 HEX) If ullNumberOfBytesToAdd is 0: E_INVALIDARG If the capacity for LUN extension is not sufficient: VDS_E_NOT_ENOUGH_SPACE (8004240F HEX) If LUN extension is failed: E_FAIL(80004005 HEX)

Shrink

Table 2-51 IVdsLun::Shrink

definition	HRESULT Shrink(ULONGLONG uNumberOfBytesToRemove, IVdsAsync ***ppAsync*)
description	This shrinks LUN.
parameter	[IN] ullNumberOfBytesToRemove The bytes for removement in LUN [OUT] ppAsync

	The pointer of buffer for the unsynchronized objects. It must not be NULL.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	It always returns VDS_E_NOT_SUPPORTED.
error	VDS_E_NOT_SUPPORTED (80042400 HEX)

QueryPlexes

Table 2-52 IVdsLun::QueryPlexes

definition	HRESULT QueryPlexes(IEnumVdsObject ***ppEnum*)
description	This returns the list of LUN plexes.
parameter	[OUT] ppEnum The pointer of buffer for the enumeration objects for LUN plexes. It must not be NULL.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This returns the list of LUN plexes object in the internal buffer.
error	If ppEnum is NULL: E_POINTER (80004003 hex)

AddPlex

Table 2-53 IVdsLun::AddPlex

definition	HRESULT AddPlex(VDS_OBJECT_ID *lunId*, IVdsAsync ***ppAsync*)
description	This adds LUN as the LUN plex(the duplicated LUN).
parameter	[IN] lunId The GUID of LUN that is added as the LUN plex. [OUT] ppAsync The pointer of buffer for the unsynchronized objects. It must not be NULL.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This returns the list of LUN plexes in the internal buffer.
error	If ppEnum is NULL: E_POINTER (80004003 HEX)

RemovePlex

Table 2-54 IVdsLun::RemovePlex

definition	HRESULT RemovePlex(VDS_OBJECT_ID *plexId*, IVdsAsync ***ppAsync*)
description	This removes the LUN plex in LUN.
parameter	[IN] lunId The GUID of the LUN plex that is removed in the LUN. [OUT] ppAsync The pointer of buffer for the unsynchronized objects. It must not be NULL.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This always returns VDS_E_NOT_SUPPORTED..
error	VDS_E_NOT_SUPPORTED (80042400 HEX)

Recover

Table 2-55 IVdsLun::Recover

definition	HRESULT Recover(IVdsAsync ***ppAsync*)
description	This begins to recover the LUN.
parameter	[OUT] ppAsync The pointer of buffer for the unsynchronized object. It must not be NULL.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This always returns VDS_E_NOT_SUPPORTED..
error	VDS_E_NOT_SUPPORTED (80042400 HEX)

SetMask

Table 2-56 IVdsLun::SetMask

definition	HRESULT SetMask(LPWSTR pwszUnmaskingList)
description	This sets the masking of LUN.
parameter	[IN] pwszUnmaskingList This shows the node of authorization for the LUN. This can be included in the nodes that are divided with ";". If "" is specified, the LUN cannot be accessed from any nodes. It is not supported that the LUN can be accessed from all of the nodes by specifying "*". The parameter must not be NULL.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error

implementation	(1) Disassociates HITACHI_SCSIProtocolController that is mapped for LUN. (2) Removes HITACHI_SCSIProtocolController and HITACHI_StorageHardwareID. (3) Adds the new HITACHI_SCSIProtocolController. (4) Adds HITACHI_StorageHardwareID of the nodes that is assigned with (3) (5) Maps LUN to (3).
error	If pwszUnmaskingList is NULL: E_POINTER (80004003 HEX) If pwszUnmaskingList is "*": If LUN is not mapped: VDS_E_NOT_SUPPORTED (80042400 HEX) If the masking is failed: E_FAIL(80004005 HEX)

Delete

Table 2-57 IVdsLun::Delete

definition	HRESULT Delete()
description	This removes the LUN and the LUN plex.
parameter	No parameters.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	(1) Removes the HITACHI_StorageVolume (2) Removes the HITACHI_SCSIProtocolController and HITACHI_StorageHardwareID.
error	If it failed to remove LUN: E_FAIL(80004005 HEX)

QueryHints

Table 2-58 IVdsLun::QueryHints

definition	HRESULT QueryHints(VDS_HINTS **pHints*)
description	This returns the hint that is applied for LUN.
parameter	[OUT] pHints The buffer for the hint. It must not be NULL.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This always returns VDS_E_NOT_SUPPORTED.
error	VDS_E_NOT_SUPPORTED (80042400 HEX)

ApplyHints

Table 2-59 IVdsLun::ApplyHints

definition	HRESULT ApplyHints(VDS_HINTS **pHints*)
description	This applies the hint to LUN.
parameter	[IN] pHints The hint applied to LUN.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This always returns VDS_E_NOT_SUPPORTED..
error	VDS_E_NOT_SUPPORTED (80042400 HEX)

SetStatus

Table 2-60 IVdsLun::SetStatus

definition	HRESULT SetStatus(VDS_LUN_STATUS *status*)
description	This sets the status of LUN.
parameter	[IN] status The new status of LUN.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This returns VDS_E_NOT_SUPPORTED.
error	VDS_E_NOT_SUPPORTED (80042400 HEX)

QueryMaxLunExtendSize

Table 2-61 IVdsLun::QueryMaxLunExtendSize

definition	HRESULT QueryMaxLunExtendSize(VDS_OBJECT_ID **pDriveIdArray*, LONG lNumberOfDrives, ULONGLONG *pullMaxBytesToBeAdded)
description	This returns the maximum size for LUN extension.
parameter	[IN] pDriveIdArray The array of GUID of drives for LUN extension. If lNumberOfDrives is 0, it may be NULL. If the parameter is 0, the provider needs to sum the maximum size of LUN extension for finding the valid drivers. The current provider supports the sum of maximum size of LUN extension by finding any drives. The parameter is unavailable. [IN] lNumberOfDrives The number of drives in pDriveIdArray. The current provider supports the sum of maximum size of LUN extension by finding

	any drives. The parameter is unavailable. [OUT] pullMaxBytesToBeAdded The buffer for the maximum size for LUN extension. It must not be NULL.s
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This stores the maximum size for extending LUN in the array group of subsystem and returns it.
error	If pullMaxBytesToBeAdded is NULL: E_POINTER (80004003 HEX)

2.2.12 IVdsLunControllerPorts

2.2.12.1 summary of interface

Table 2-62 IVdsLunControllerPorts

description	This supports the operation of associating LUN with port. The LUN is equal to StorageVolume of CIM provider.
implementation	CHdvmLun

2.2.12.2 specification of method

AssociateControllerPorts

Table 2-63 IVdsLunControllerPorts::AssociateControllerPorts

definition	HRESULT AssociateControllerPorts(VDS_OBJECT_ID *pActiveControllerPortIdArray, LONG lNumberOfActiveControllerPorts, VDS_OBJECT_ID *pInactiveControllerPortIdArray, LONG lNumberOfInactiveControllerPorts)
description	This sets the active controller port or the inactive controller port for the LUN.
parameter	[IN] pActiveControllerPortIdArray The array of GUID for the active controller. If lNumberOfActiveControllerPorts is not 0, it must not be NULL. [IN] lNumberOfActiveControllerPorts The number of controller ports in pActiveContollerPortIdArray [IN] pInactiveControllerPortIdArray The array of GUID for the inactive controller ports. If lNumberOfInactiveControllerPorts is not 0, it must not be NULL. [IN] lNumberOfInactiveControllerPorts The number of controller ports in pInactiveContollerPortIdArray.
return value	Normal termination: S_OK (0)

	Abnormal termination: status code of error
implementation	This finds the paths that are associated with the LUN in the internal buffer and changes the status of controller port to the active or inactive one.
error	If the specified GUID does not exist: VDS_E_OBJECT_NOT_FOUND (80042405 HEX) If the operation of mapping or unmapping is failed: E_FAIL(80004005 HEX) If lNumberOfActiveControllerPorts>0 and pActiveControllerPortIdArray is NULL: E_POINTER (80004003 HEX)

QueryActiveControllerPorts

Table 2-64 IVdsLunControllerPorts::QueryActiveControllerPorts

definition	HRESULT QueryActiveControllerPorts(IEnumVdsObject ***ppEnum*)
description	This returns the list of the active controller ports of LUN.
parameter	[OUT] ppEnum The pointer of buffer for the enumeration objects of the active controller ports of the LUN. It must not be NULL.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This finds the paths that are associated with the LUN in information of paths that are stored in the internal buffer and returns the list of associated controller ports.
error	If ppEnum is NULL: E_POINTER (80004003 HEX)

2.2.13 IVdsLunNaming

2.2.13.1 summary of interface

Table 2-65 IVdsLunNaming

description	This supports the operation of setting the name of LUN.
implementation	CHdvmLun

2.2.13.2 specification of method

SetFriendlyName

Table 2-66 IVdsLunNaming::SetFriendlyNames

definition	HRESULT SetFriendlyName(LPWSTR *pwszName*)
description	This sets the name of LUN.
parameter	[IN] pwszName The name of LUN. The maximum length of string is 64 characters with the end termination in Unicode. It must not be NULL.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This calls HITACHI_StorageVolume.SetProperty(ElementName).
error	If pwszName is NULL: E_POINTER (80004003 HEX) If failed: E_FAIL(80004005 HEX)

2.2.14 IVdsLunPlex

2.2.14.1 summary of interface

Table 2-67 IVdsLunPlex

description	This supports the operation of LUN plex.
implementation	CHdvmLunPlex

2.2.14.2 specification of method

GetProperties

Table 2-68 IVdsLunPlex::GetProperties

definition	HRESULT GetProperties(VDS_LUN_PLEX_PROP *pPlexProp)
description	This returns the property of LUN plex.
parameter	[OUT] pPlexProp The buffer for the property of LUN plex. It must not be NULL.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This returns the structure VDS_LUN_PLEX_PROP that is stored in the internal buffer. Cf.2.3.7
error	If pPlexProp is NULL: E_POINTER (80004003 hex)

GetLun

Table 2-69 IVdsLunPlex::GetLun

definition	HRESULT GetLun(IVdsLun **ppLun)
description	This returns the LUN that LUN plex belongs to.
parameter	[OUT] ppLun The pointer of buffer for LUN object that has the LUN plex. It must not be NULL.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This returns the LUN object that is stored in the internal buffer.
error	If ppLun is NULL: E_POINTER (80004003 hex)

QueryExtents

Table 2-70 IVdsLunPlex::QueryExtents

definition	HRESULT QueryExtents(VDS_DRIVE_EXTENT **ppExtentArray, LONG *plNumberOfExtents)
description	This returns the array of extents of drives that create LUN plex.
parameter	[OUT] ppExtentArray The pointer of buffer for the array of drive extents that create the

	LUN plex. It must not be NULL. [OUT] plNumberOfExtents The buffer for the number of drive extents in ppExtentArray. It must not be NULL.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This acquires the extents in the internal buffer of drives that create the LUN plex.
error	If ppExtentArray or plNumberOfExtents is NULL: E_POINTER (80004003 hex)

QueryHints

Table 2-71 IVdsLunPlex::QueryHints

definition	HRESULT QueryHints(VDS_HINTS **pHints*)
description	This returns the hint applied to LUN plex.
parameter	[OUT] pHints The buffer for the hint. It must not be NULL.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This always returns VDS_E_NOT_SUPPORTED.
error	VDS_E_NOT_SUPPORTED (80042400 HEX)

ApplyHints

Table 2-72 IVdsLunPlex::ApplyHints

definition	HRESULT ApplyHints(VDS_HINTS **pHints*)
description	This sets the hint to LUN plex.
parameter	[IN] pHints The hint for applying to LUN plex.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This always returns VDS_E_NOT_SUPPORTED.
error	VDS_E_NOT_SUPPORTED (80042400 HEX)

2.2.15 IVdsDrive

2.2.15.1 summary of interface

Table 2-73 IVdsDrive

description	This supports the operation of drives.
implementation	CHdvmDrive

2.2.15.2 specification of method

ClearFlags

Table 2-74 IVdsDrive:: ClearFlags

definition	HRESULT ClearFlags (ULONG *ulFlags*)
description	This cleans the flag of drive.
parameter	[IN] ulFlags The flag for cleaning.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This always returns VDS_E_NOT_SUPPORTED.
error	VDS_E_NOT_SUPPORTED (80042400 HEX)

GetProperties

Table 2-75 IvdsDrive::GetProperties

definition	HRESULT GetProperties(VDS_DRIVE_PROP **pDriveProp*)
description	This returns the property of drive.
parameter	[OUT] pDriveProp The buffer for the property of drive. It must not be NULL.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This returns the structure VDS_DRIVE_PROP in the internal buffer. Cf:2.3.8
error	If pDriveProp is NULL: E_POINTER (80004003 HEX)

GetSubSystem

Table 2-76 IVdsDrive::GetSubSystem

definition	HRESULT GetSubSystem(IVdsSubSystem ***ppSubSystem*)

description	This returns the subsystem that the drive belong to.
parameter	[OUT] ppSubSystem The pointer of buffer for the subsystem object that has the drive. It must not be NULL.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This returns the subsystem object in the internal buffer.
error	If ppSubSystem is NULL: E_POINTER (80004003 HEX)

QueryExtents

Table 2-77 IVdsDrive::QueryExtents

definition	HRESULT QueryExtents(VDS_DRIVE_EXTENT ***ppExtentArray*, LONG *plNumberOfExtents)
description	This returns the list of drive extents.
parameter	[OUT] ppExtentArray The pointer of buffer for the list of extents. It must not be NULL. [OUT] plNumberOfExtent The pointer of buffer for the number of extents. It must not be NULL.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This returns the extents in the internal buffer. Cf:2.3.9
error	If ppExtentArray and plNumberOfExtent are NULL: E_POINTER (80004003 HEX)

SetFlags

Table 2-78 IVdsDrive::SetFlags

definition	HRESULT SetFlags (ULONG *ulFlags*)
description	This sets the flag of drive.
parameter	[IN] ulFlags The flag of drive.
return value	VDS_E_NOT_SUPPORTED (80042400 HEX)
implementation	This always returns VDS_E_NOT_SUPPORTED.
error	VDS_E_NOT_SUPPORTED (80042400 HEX)

SetStatus

Table 2-79 IVdsDrive::SetStatus

definition	HRESULT SetStatus(VDS_PORT_STATUS *status*)
description	This sets the status of port.
parameter	[IN] status The new status of port.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This always returns VDS_E_NOT_SUPPORTED.
error	VDS_E_NOT_SUPPORTED (80042400 HEX)

2.2.16 IEnumVdsObject

2.2.16.1 summary of interface

Table 2-80 IEnumVdsObject

description	This supports the operation of enumeration of the VDS objects that have the specified type.
implementation	CHdvmEnumerator

2.2.16.2 specification of method

Next

Table 2-81 IEnumVdsObject::Next

definition	HRESULT Next(
	  ULONG *celt*,
	  IUnknown **ppObjectArray,
	  ULONG *pcFetched
	)

description	This returns the next object.
parameter	[IN] celt The number of the received object. [OUT] ppObjectArray The pointer of buffer for the next object. It must not be NULL. [OUT] pcFetched The buffer for the number of the objects that succeeded to be received. It must not be NULL.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This returns the next object in the internal list.
error	If ppObjectArray or pcFetched is NULL:

	E_POINTER (80004003 hex) If the number of the enumerated objects is less than the number of the objects that are specified by celt: S_FALSE

Skip

Table 2-82 IEnumVdsObject::Skip

definition	HRESULT Skip(ULONG *celt*)
description	This skips the next object.
parameter	[IN] celt The number of the skipped objects.
return value	Normal termination: S_OK (0) If the number of the latter objects is less than the number of objects that are specified by celt: S_FALSE (1) Abnormal termination: status code of error
implementation	This skips the next object in the internal list.
error	If the number of the latter objects is less than the number of objects that are specified by celt: S_FALSE

Reset

Table 2-83 IEnumVdsObject::Reset

definition	HRESULT Reset()
description	This resets the operation of enumeration and returns the first object.
parameter	No parameters.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This resets the internal list.
error	No error.

Clone

Table 2-84 IEnumVdsObject::Clone

definition	HRESULT Clone(IEnumVdsObject ***ppEnum*)
description	This creates the clone of enumeration objects.
parameter	[OUT] ppEnum

	The pointer of buffer for the clone of the current enumeration objects. It must not be NULL.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This always returns VDS_E_NOT_SUPPORTED.
error	VDS_E_NOT_SUPPORTED (80042400 HEX)

2.2.17 IVdsAsync

2.2.17.1 summary of interface

Table 2-85 IVdsAsync

description	This supports the unsynchronized operation.
implementation	CHdvmAsyncOperation

2.2.17.2 specification of method

Cancel

Table 2-86 IVdsAsync::Cancel

definition	HRESULT Cancel()
description	This cancels the unsynchronized operation. The provider of the current version does not execute the cancel operation and returns VDS_E_CANCEL_TOO_LATE.
parameter	No parameters.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This returns the error code with the value(running/completed) in the internal buffer that the thread for polling the status of the HITACHI_ConcreateJob has.
error	If the unsynchronized operation is running: VDS_E_CANCEL_TOO_LATE If the unsynchronized operation is completed: VDS_E_OPERATION_COMPLETED

QueryStatus

Table 2-87 IVdsAsync::QueryStatus

definition	HRESULT QueryStatus(HRESULT **pHrResult*, ULONG *pulPercentCompleted

	)
description	This returns the status of the unsynchronized operation.
parameter	[OUT] pHrResult The pointer of buffer for the status of the unsynchronized operation. It must not be NULL. If the operation is running, VDS_E_OPERATION_PENDING. If the operation is completed normally, S_OK. If it is failed, E_FAIL [OUT] pulPercentCompleted The pointer of buffer for the status of processing of the unsynchronized operation. It must not be NULL. If the operation is running, the provider stores 0. If it is failed, the provider stores 100.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This returns the status of running or termination with the value in the internal buffer that the thread for polling the status of the HITACHI_ConcreateJob has.
error	If pHrResult or pulPercentCompleted is NULL: E_POINTER (80004003 HEX) (If the method is completed normally, this returns S_OK.)

Wait

Table 2-88 IVdsAsync::Wait

definition	HRESULT Wait(HRESULT **pHrResult*, VDS_ASYNC_OUTPUT **pAsyncOut*)
description	This returns after the unsynchronized operation is completed.
parameter	[OUT] pHrResult The pointer of buffer for the status of the unsynchronized operation. It must not be NULL. If the operation is completed normally, this returns S_OK, and if it is failed, this returns E_FAIL. [OUT] pAsyncOut The pointer of buffer for the status of the unsynchronized operation. It must not be NULL. Cf.2.3.10.
return value	Normal termination: S_OK (0) Abnormal termination: status code of error
implementation	This returns the status of the operation after the thread for polling the status of the HITACHI_ConcreateJob is completed.
error	If pHrResult or pAsyncOut is NULL: E_POINTER (80004003 HEX)

2.2.18 IVdsProviderSupport

2.2.18.1 summary of interface

Table 2-89 IVdsProviderSupport

description	This supports the version of hardware provider and software provider. It can be used by the VDS client.
implementation	CHdvmProvider

2.2.18.2 specification of method

GetVersionSupport

Table 2-90 IVdsProviderSupport::GetVersionSupport

definition	HRESULT GetVersionSupport(ULONG* ulVersionSupport)
description	This returns the version of the supported VDS.
parameter	No parameters.
return value	VDS_VFS_1_1(fixed value)
implementation	This always returns VDS_VFS_1_1.
error	No error.

2.3 Structure of HDvM VDS provider

2.3.1 VDS_PROVIDER_PROP

2.3.1.1 summary

Table 2-91 VDS_PROVIDER_PROP

description	This defines the property of the provider.
CIM object	None

2.3.1.2 specification of the fields

id

Table 2-92 Id of VDS_PROVIDER_PROP

Type	VDS_OBJECT_ID
description	The eternal GUID of the provider

Value	The fixed value by the utility of UUIDGEN.EXE {D0174D89-27F4-4155-BAA2-5A05F5F9AE84}

pwszName

Table 2-93 pwszName of VDS_PROVIDER_PROP

Type	LPWSTR
description	The name of provider.
Value	"Device Manager VDS Provider"

guidVersionId

Table 2-94 guidVersionId of VDS_PROVIDER_PROP

Type	GUID
description	GUID of the version of the provider
Value	The fixed value by UUIDGEN.EXE {3D590AB5-73AD-419A-8BE5-8E44AEBAF133}

pwszVersion

Table 2-95 pwszVersion of VDS_PROVIDER_PROP

Type	LPWSTR
description	The version of the provider.
Value	"1.1"

type

Table 2-96 type of VDS_PROVIDER_PROP

Type	VDS_PROVIDER_TYPE (the type of enum)
description	The type of the provider.
Value	VDS_PT_HARDWARE

ulFlags

Table 2-97 ulFlags of VDS_PROVIDER_PROP

Type	ULONG
description	The flag of the provider.
Value	0

ulStripeSizeFlags

Table 2-98 ulStripeSizeFlags of VDS_PROVIDER_PROP

Type	ULONG
description	The size of stripe that is used only by software provider.
Value	0

sRebuildPriority

Table 2-99 sRebuildPriority of VDS_PROVIDER_PROP

Type	ULONG
description	The priority of rebuild that is used only by software provider.
Value	0

2.3.2 VDS_SUB_SYSTEM_PROP

2.3.2.1 summary

Table 2-100 VDS_SUB_SYSTEM_PROP

description	This defines the property of subsystem.
CIM object	HITACHI_StorageSystem

2.3.2.2 specification of the fields

id

Table 2-101 id of VDS_SUB_SYSTEM_PROP

Type	VDS_OBJECT_ID
description	GUID of subsystem
Value	GUID that is generated for each received information dynamically.

pwszFriendlyName

Table 2-102 pwszFriendlyName of VDS_SUB_SYSTEM_PROP

Type	LPWSTR
description	The name of subsystem
Value	HITACHI_StorageSystem.ElementName

pwszIdentification

Table 2-103 pwszIdentification of VDS_SUB_SYSTEM_PROP

Type	LPWSTR
description	ID of subsystem
Value	HITACHI_StorageSystem.Name

ulFlags

Table 2-104 ulFlags of VDS_SUB_SYSTEM_PROP

Type	ULONG	
description	The flag of subsystem	
Value	VDS_SF_LUN_MASKING_CAPABLE	0x1
	VDS_SF_LUN_NAMING_CAPABLE	0x100
	VDS_SF_SUPPORTS_STRIPE_LUNS	0x2000
	VDS_SF_SUPPORTS_MIRROR_LUNS	0x4000
	VDS_SF_SUPPORTS_PARITY_LUNS	0x8000

ulStripeSizeFlags

Table 2-105 ulStripeSizeFlags of VDS_SUB_SYSTEM_PROP

Type	ULONG
description	The size of stripe that is supported on the subsystem.
Value	DF700H: 64k, 512k DF except for DF700H: 64k RAID: 512k

status

Table 2-106 status of VDS_SUB_SYSTEM_PROP

Type	VDS_SUB_SYSTEM_STATUS
description	The status of the subsystem
Value	VDS_SSS_ONLINE

health

Table 2-107 headlth of VDS_SUB_SYSTEM_PROP

Type	VDS_HEALTH (the type of enum)
description	The status of health of subsystem
Value	VDS_H_HEALTHY

sNumberOfInternalBuses

Table 2-108 sNumberOfInternalBuses of VDS_SUB_SYSTEM_PROP

Type	SHORT
description	The number of buses
Value	1

sMaxNumberOfSlotsEachBus

Table 2-109 sMaxNumberOfSlotsEachBus of VDS_SUB_SYSTEM_PROP

Type	SHORT
description	The maximum number of slots in the bus
Value	The number of HITACHI_DiskDirve associated with HITACHI_StorageSystem and CIM_SystemDevice.

sMaxNumberOfControllers

Table 2-110 sMaxNumberOfControllers of VDS_SUB_SYSTEM_PROP

Type	SHORT
description	The maximum number of controllers.
Value	The number of HITACHI_StorageProcessorSystem associated with HITACHI_StorageSystem and CIM_ComponentCS.

sRebuildPriority

Table 2-111 sRebuildPriority of VDS_SUB_SYSTEM_PROP

Type	SHORT
description	The priority of rebuild of LUN
Value	0

2.3.3 VDS_CONTROLLER_PROP

2.3.3.1 summary

Table 2-112 VDS_CONTROLLER_PROP

description	This defines the property of controller.
CIM object	HITACHI_StorageProcessorSystem

2.3.3.2 specification of fields

id

Table 2-113 id of VDS_CONTROLLER_PROP

Type	VDS_OBJECT_ID
description	GUID of controller
Value	GUID that is generated for all information dynamically.

pwszFriendlyName

Table 2-114 pwszFriendlyName of VDS_CONTROLLER_PROP

Type	LPWSTR
description	The name of controller
Value	HITACHI_StorageProcessorSystem.ElementName

pwszIdentification

Table 2- 115 pwszIdentification of VDS_CONTROLLER_PROP

Type	LPWSTR
description	ID of controller
Value	HITACHI_StorageProcessorSystem.Name

status

Table 2- 116 status of VDS_CONTROLLER_PROP

Type	VDS_CONTROLLER_STATUS
description	The status of controller
Value	VDS_SSS_ONLINE

health

Table 2-117 health of VDS_CONTROLLER_PROP

Type	VDS_HEALTH
description	The status of health of controller
Value	VDS_H_HEALTHY

sNumberOfPort

Table 2- 118 sNumberOfPort of VDS_CONTROLLER_PROP

Type	SHORT
description	The number of ports
Value	The number of HITACHI_FCPort associated with HITACHI_StorageProcessorSystem.

2.3.4 VDS_PORT_PROP

2.3.4.1 structure summary

Table 2-119 VDS_PORT_PROP

description	This defines the property of ports.
CIM object	HITACHI_FCPort

2.3.4.2 specification of fields

id

Table 2-120 id of VDS_PORT_PROP

Type	VDS_OBJECT_ID
description	GUID of port
Value	GUID that is generated for each received information dynamically

pwszFriendlyName

Table 2-121 pwszFriendlyName of VDS_PORT_PROP

Type	LPWSTR
description	The name of port
Value	HITACHI_FCPort.ElementName

pwszIdentification

Table 2-122 pwszIdentification of VDS_PORT_PROP

Type	LPWSTR
description	ID of port
Value	HITACHI_FCPort.DeviceID

status

Table 2-123 status of VDS_PORT_PROP

Type	VDS_PORT_STATUS (enum)
description	The status of port
Value	VDS_SSS_ONLINE

2.3.5 VDS_LUN_PROP

2.3.5.1 structure summary

Table 2-124 VDS_LUN_PROP

description	This defines the property of LUN.
CIM object	HITACHI_StorageVolume

2.3.5.2 specification of fields

id

Table 2-125 id of VDS_LUN_PROP

Type	VDS_OBJECT_ID
description	GUID of LUN
Value	GUID that is generated for each received information dynamically

ullSize

Table 2-126 ullSize of VDS_LUN_PROP

Type	ULONGLONG
description	The capacity of LUN (Bytes)
Value	This sets the capacity with the value of BlockSize * NumberOfBlocks by getting the property of BlockSize and NumberOfBlocks of HITACHI_StorageVolume.

pwszFriendlyName

Table 2-127 pwszFriendlyName of VDS_LUN_PROP

Type	LPWSTR
description	The name of LUN
Value	HITACHI_StorageVolume.ElementName

pwszIdentification

Table 2-128 pwszIdentification of VDS_LUN_PROP

Type	LPWSTR
description	ID of LUN
Value	HITACHI_StorageVolume.DeviceID

pwszUnmaskingList

Table 2-129 pwszUnmaskingList of VDS_LUN_PROP

Type	LPWSTR
description	The list of the accessible machines
Value	The list of WWN divided with ";".

ulFlags

Table 2-130 ulFlags of VDS_LUN_PROP

Type	ULONG
description	The flag of LUN
Value	0

type

Table 2-131 type of VDS_LUN_PROP

Type	VDS_LUN_TYPE (the type of enum)
description	The type of LUN
Value	It is based on the level of RAID in HITACHI_StorageVolume.ErrorMethodology. If RAID0: VDS_LT_STRIPE If RAID1,0+1: VDS_LT_MIRROR If RAID3,4,5,6: VDS_LT_PARITY

status

Table 2-132 status of VDS_LUN_PROP

Type	VDS_LUN_STATUS (the type of enum)
description	The status of LUN
Value	VDS_SSS_ONLINE

health

Table 2-133 health of VDS_LUN_PROP

Type	VDS_HEALTH (the type of enum)
description	The status of health of LUN
Value	VDS_H_HEALTHY

TransitionState

Table 2-134 TransitionState of VDS_LUN_PROP

Type	VDS_TRANSITION_STATE (the type of enum)
description	The status of configuration of LUN
Value	VDS_SSS_ONLINE

sRebuildPriority

Table 2-135 sRebuildPriority of VDS_LUN_PROP

Type	SHORT
description	The priority of rebuild of LUN
Value	0

2.3.6 VDS_LUN_INFORMATION

2.3.6.1 structure summary

Table 2-136 VDS_LUN_INFORMATION

description	This defines the information of LUN.
CIM object	HITACHI_StorageVolume

2.3.6.2 specification of fields

version

Table 2-137 version of VDS_LUN_INFORMATION

Type	ULONG
description	The version of structure
Value	VER_VDS_LUN_INFORMATION

DeviceType

Table 2-138 DeviceType of VDS_LUN_INFORMATION

Type	BYTE

description	The SCSI2 type of device
Value	0

DeviceTypeModifier

Table 2-139 DeviceTypeModifier of VDS_LUN_INFORMATION

Type	BYTE
description	The modifier of the SCSI2device type
Value	0

bCommandQueueing

Table 2-140 bCommandQueueing of VDS_LUN_INFORMATION

Type	BOOL
description	The flag of queue for command
Value	false

BusType

Table 2-141 BusType of VDS_LUN_INFORMATION

Type	VDS_STORAGE_BUS_TYPE (type of enum)
description	The type of bus
Value	VDSBusTypeFibre

szVendorId

Table 2-142 szVendorId of VDS_LUN_INFORMATION

Type	char*
description	ID of vendor
Value	HITACHI

szProductId

Table 2-143 szProductId of VDS_LUN_INFORMATION

Type	char*
description	ID of product
Value	""

szProductRevision

Table 2-144 szProductRevision of VDS_LUN_INFORMATION

Type	char*
description	The revision of product
Value	""

szSerialNumber

Table 2-145 szSerialNumber of VDS_LUN_INFORMATION

Type	char*
description	The number of serial
Value	""

diskSignature

Table 2-146 diskSignature of VDS_LUN_INFORMATION

Type	GUID
description	signature
Value	0

deviceIdDescriptor

Table 2-147 deviceIdDescriptor of VDS_LUN_INFORMATION

Type	VDS_STORAGE_DEVICE_ID_DESCRIPTOR
description	The descriptor of LUN
Value	0

cInterconnects

Table 2-148 cInterconnects of VDS_LUN_INFORMATION

Type	ULONG
description	The number of ports of interconnects
Value	0

rgInterconnects

Table 2-149 rgInterconnects of VDS_LUN_INFORMATION

Type	VDS_INTERCONNECT
description	The port of interconnect
Value	0

2.3.7 VDS_LUN_PLEX_PROP

2.3.7.1 structure summary

Table 2-150 VDS_LUN_PLEX_PROP

description	This defines the property of LUN plex.
CIM object	The instance of HITACHI_StorageVolume associated with LUN and HITACHI_StorageSynchronized.

2.3.7.2 specification of fields

id

Table 2-151 id of VDS_LUN_PLEX_PROP

Type	VDS_OBJECT_ID
description	GUID of LUN plex
Value	GUID that is generated for each received information dynamically

ullSize

Table 2-152 ullSize of VDS_LUN_PLEX_PROP

Type	ULONGLONG
description	The capacity of LUN plex (Bytes)
Value	This sets the capacity with the value of BlockSize * NumberOfBlocks by getting the property of BlockSize and NumberOfBlocks of HITACHI_StorageVolume.

type

Table 2-153 type of VDS_LUN_PLEX_PROP

Type	VDS_LUN_PLEX_TYPE (enumType)
description	The type of LUN plex
Value	It is based on the level of RAID in HITACHI_StorageVolume.ErrorMethodology. If RAID0: VDS_LT_STRIPE If RAID1,0+1: VDS_LT_MIRROR If RAID3,4,5,6: VDS_LT_PARITY

status

Table 2-154 status of VDS_LUN_PLEX_PROP

Type	VDS_LUN_PLEX_STATUS (type of enum)
description	The status of LUN plex
Value	VDS_SSS_ONLINE

health

Table 2-155 health of VDS_LUN_PLEX_PROP

Type	VDS_HEALTH (type of enum)
description	The status of health of LUN plex
Value	VDS_H_HEALTHY

TransitionState

Table 2-156 TransitionState of VDS_LUN_PLEX_PROP

Type	VDS_TRANSITION_STATE (type of enum)
description	The status of configuration of LUN plex
Value	VDS_SSS_ONLINE

ulFlags

Table 2-157 ulFlags of VDS_LUN_PLEX_PROP

Type	ULONG
description	The flag of LUN plex
Value	0

ulStripeSize

Table 2-158 ulStripeSize of VDS_LUN_PLEX_PROP

Type	ULONG
description	The size of stripe (Bytes)
Value	0

sRebuildPriority

Table 2-159 sRebuildPriority of VDS_LUN_PLEX_PROP

Type	SHORT
description	The priority of rebuild of LUN plex
Value	0

2.3.8 VDS_DRIVE_PROP

2.3.8.1 structure summary

Table 2-160 VDS_DRIVE_PROP

description	This defines the property of drive.
CIM object	HITACHI_DiskDrive

2.3.8.2 specification of fields

id

Table 2-161 id of VDS_DRIVE_PROP

Type	VDS_OBJECT_ID
description	GUID of drive
Value	GUID that is generated for each received information dynamically

ullSize

Table 2-162 ullSize of VDS_DRIVE_PROP

Type	ULONGLONG
description	The size of drive (bytes)
Value	HITACHI_DiskDrive.MaxMediaSize * 1000

pwszFriendlyName

Table 2-163 pwszFriendlyName of VDS_DRVIE_PROP

Type	LPWSTR
description	The name of drive
Value	HITACHI_DiskDrive.ElementName

pwszIdentification

Table 2-164 pwszIdentification of VDS_DRIVE_PROP

Type	LPWSTR
description	ID of drive
Value	HITACHI_DiskDrive.DeviceID

uFlags

Table 2-165 uFlags of VDS_DRIVE_PROP

Type	ULONG
description	The flag of drive
Value	0

status

Table 2-166 status of VDS_DRIVE_PROP

Type	VDS_DRIVE_STATUS
description	The status of running drive.
Value	VDS_SSS_ONLINE.

health

Table 2-167 health of VDS_DRIVE_PROP

Type	VDS_HEALTH
description	The health of drive
Value	VDS_H_HEALTHY

sInternalBusNumber

Table 2-168 sInternalBusNumber of VDS_DRIVE_PROP

Type	SHORT
description	The number of bus of drive
Value	0

sSlotNumber

Table 2-169 sSlotNumber of VDS_DRIVE_PROP

Type	SHORT
description	The number of slot of drive
Value	HITACHI_DiskDrive.DeviceID

2.3.9 VDS_DRIVE_EXTENT

2.3.9.1 structure summary

Table 2-170 VDS_DRIVE_EXTENT

description	This defines the detail of drive extent.
CIM object	HITACHI_DiskExtent

2.3.9.2 specification of structure field

id

Table 2-171 id of VDS_DRIVE_EXTENT

Type	VDS_OBJECT_ID
description	GUID of the drive
Value	GUID of drive that is generated for each received information dynamically

LunId

Table 2-172 LunId of VDS_DRIVE_EXTENT

Type	VDS_OBJECT_ID
description	GUID of LUN associated with the drive.
Value	GUID of LUN

ullSize

Table 2-173 ullSize of VDS_DRIVE_EXTENT

Type	ULONGLONG
description	The size of extent (bytes)
Value	HITACHI_DiskExtent.BlockSize is multiplied by HITACHI_DiskExtent.ConsumableBlocks.

bUsed

Table 2-174 bUsed of VDS_DRIVE_EXTENT

Type	BOOL
description	If it is associated with LUN, TRUE. If it is not, FALSE.
Value	True or False

2.3.10 VDS_ASYNC_OUTPUT

2.3.10.1 structure summary

Table 2-175 VDS_ASYNC_OUTPUT

description	This defines the result of the unsynchronized operation.
CIM object	HITACHI_ConcreteJob,HITACHI_StorageVolume

2.3.10.2 specification of fields

type

Table 2-176 type of VDS_ASYNC_OUTPUT

Type	VDS_ASYNC_OUTPUT_TYPE
description	This shows the type of the unsynchronized operation.
Value	If a new LUN is created(IVdsSubSystem::CreateLun): VDS_ASYNCOUT_CREATELUN If LUN is extended(IVdsLun::Extend): VDS_ASYNCOUT_EXTENDLUN

union

The second member of this structure is union. It is set with the followings based on the type.

(1) structure cl

Table 2-177 cl.pLunUnk of VDS_ASYNC_OUTPUT

Type	IUnknown*
description	The new object of LUN
Value	The LUN object that is equal to generated HITACHI_StorageVolume

(2) structure **

Table 2-178 **.pLunUnk of VDS_ASYNC_OUTPUT

Type	IUnknown*
description	The extended LUN object.
Value	The LUN object that is equal to the extended HITACHI_StorageVolume.

Acronyms and Abbreviations

CIM	Common Information Model
HDvM	HiCommand Device Manager
LUN	Logical Unit Number
SAN	Storage Area Network
VDS	Virtual Disk Service