[MS-WDSMA]:
Windows Deployment Services Multicast Application Protocol

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation for protocols, file formats, languages, standards as well as overviews of the interaction among each of these technologies.
- **Copyrights.** This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you may make copies of it in order to develop implementations of the technologies described in the Open Specifications and may distribute portions of it in your implementations using these technologies or your documentation as necessary to properly document the implementation. You may also distribute in your implementation, with or without modification, any schema, IDL's, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications.
- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.
- **Patents.** Microsoft has patents that may cover your implementations of the technologies described in the Open Specifications. Neither this notice nor Microsoft's delivery of the documentation grants any licenses under those or any other Microsoft patents. However, a given Open Specification may be covered by Microsoft Open Specification Promise or the Community Promise. If you would prefer a written license, or if the technologies described in the Open Specifications are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **Trademarks.** The names of companies and products contained in this documentation may be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights. For a list of Microsoft trademarks, visit www.microsoft.com/trademarks.
- **Fictitious Names.** The example companies, organizations, products, domain names, email addresses, logos, people, places, and events depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

Reservation of Rights. All other rights are reserved, and this notice does not grant any rights other than specifically described above, whether by implication, estoppel, or otherwise.

Tools. The Open Specifications do not require the use of Microsoft programming tools or programming environments in order for you to develop an implementation. If you have access to Microsoft programming tools and environments you are free to take advantage of them. Certain Open Specifications are intended for use in conjunction with publicly available standard specifications and network programming art, and assumes that the reader either is familiar with the aforementioned material or has immediate access to it.

Copyright © 2013 Microsoft Corporation.

Revision Summary

Date	Revision History	Revision Class	Comments
04/10/2009	0.1	Major	First Release.
05/22/2009	0.1.1	Editorial	Revised and edited the technical content.
07/02/2009	0.1.2	Editorial	Revised and edited the technical content.
08/14/2009	0.1.3	Editorial	Revised and edited the technical content.
09/25/2009	0.1.4	Editorial	Revised and edited the technical content.
11/06/2009	0.1.5	Editorial	Revised and edited the technical content.
12/18/2009	0.2	Minor	Updated the technical content.
01/29/2010	0.2.1	Editorial	Revised and edited the technical content.
03/12/2010	0.3	Minor	Updated the technical content.
04/23/2010	0.3.1	Editorial	Revised and edited the technical content.
06/04/2010	0.3.2	Editorial	Revised and edited the technical content.
07/16/2010	0.3.2	No change	No changes to the meaning, language, or formatting of the technical content.
08/27/2010	0.3.2	No change	No changes to the meaning, language, or formatting of the technical content.
10/08/2010	0.3.2	No change	No changes to the meaning, language, or formatting of the technical content.
11/19/2010	0.3.2	No change	No changes to the meaning, language, or formatting of the technical content.
01/07/2011	0.3.2	No change	No changes to the meaning, language, or formatting of the technical content.
02/11/2011	0.3.2	No change	No changes to the meaning, language, or formatting of the technical content.
03/25/2011	0.3.2	No change	No changes to the meaning, language, or formatting of the technical content.
05/06/2011	0.3.2	No change	No changes to the meaning, language, or formatting of the technical content.
06/17/2011	0.4	Minor	Clarified the meaning of the technical content.
09/23/2011	0.4	No change	No changes to the meaning, language, or formatting of the technical content.
12/16/2011	1.0	Major	Significantly changed the technical content.

Copyright © 2013 Microsoft Corporation.

Date	Revision History	Revision Class	Comments
03/30/2012	1.0	No change	No changes to the meaning, language, or formatting of the technical content.
07/12/2012	1.0	No change	No changes to the meaning, language, or formatting of the technical content.
10/25/2012	1.0	No change	No changes to the meaning, language, or formatting of the technical content.
01/31/2013	1.0	No change	No changes to the meaning, language, or formatting of the technical content.
08/08/2013	2.0	Major	Significantly changed the technical content.

Copyright © 2013 Microsoft Corporation.

Contents

Copyright © 2013 Microsoft Corporation.

Copyright © 2013 Microsoft Corporation.

1 Introduction

WDS Multicast Application Protocol is a single server, multiple client protocol.

The protocol uses WDS Multicast Transport Protocol for transmission of content to multiple clients. The protocol relies on services provided by the WDS Multicast Transport Protocol to ensure all pieces of content are delivered to all clients in a multicast session.

The protocol allows clients to join the multicast session at any point during the lifetime of the multicast session, and still be able to get all pieces of the content.

Sections 1.8, 2, and 3 of this specification are normative and can contain the terms MAY, SHOULD, MUST, MUST NOT, and SHOULD NOT as defined in RFC 2119. Sections 1.5 and 1.9 are also normative but cannot contain those terms. All other sections and examples in this specification are informative.

1.1 Glossary

The following terms are specific to this document:

Multicast: The ability of a transport protocol, such as User Datagram Protocol (UDP), to deliver messages to a group of recipients simultaneously without duplication of message unless the link to recipients split.

Block Number: The content is divided into equal blocks. The first block is assigned a value of 1, and each successive block is incrementally assigned the next higher value.

MAY, SHOULD, MUST, SHOULD NOT, MUST NOT: These terms (in all caps) are used as described in [RFC2119]. All statements of optional behavior use either MAY, SHOULD, or SHOULD NOT.

1.2 References

References to Microsoft Open Specifications documentation do not include a publishing year because links are to the latest version of the documents, which are updated frequently. References to other documents include a publishing year when one is available.

A reference marked "(Archived)" means that the reference document was either retired and is no longer being maintained or was replaced with a new document that provides current implementation details. We archive our documents online [Windows Protocol].

1.2.1 Normative References

We conduct frequent surveys of the normative references to assure their continued availability. If you have any issue with finding a normative reference, please contact dochelp@microsoft.com. We will assist you in finding the relevant information. Please check the archive site, http://msdn2.microsoft.com/en-us/library/E4BD6494-06AD-4aed-9823-445E921C9624, as an additional source.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997, http://www.rfc-editor.org/rfc/rfc2119.txt

Copyright © 2013 Microsoft Corporation.

1.2.2 Informative References

None.

1.3 Overview

WDS Multicast Application Protocol uses WDS Multicast Transport Protocol to deliver all pieces of the content to all clients in a multicast session.

When the first client joins the multicast session, WDS Multicast Transport Protocol notifies the WDS Multicast Application Protocol using a Trigger. After such a Trigger is received, the WDS Multicast Application Protocol uses the steps below to ensure delivery of all pieces of content to the clients in the multicast session:

1. The WDS Multicast Application Protocol server sends packets using the WDS Multicast Transport Protocol to query all clients for the block ranges that each client is missing from the content. WDS Multicast Application Protocol on each client, on receiving such a packet, sends a reply back to the server with the list of block ranges that the client is missing. WDS Multicast Application Protocol on the server-side receives all replies from clients via WDS Multicast Transport Protocol.
2. WDS Multicast Application Protocol on the server-side sends the missing pieces to all clients using WDS Multicast Transport Protocol.
3. After all missing pieces have been transmitted, WDS Multicast Application Protocol on the server starts over from step 1.

1.4 Relationship to Other Protocols

WDS Multicast Application Protocol uses WDS Multicast Transport Protocol as its transport to send the pieces of content to all clients in the multicast session. The following diagram specifies the relationship on WDS Multicast Application Protocol with other protocols:

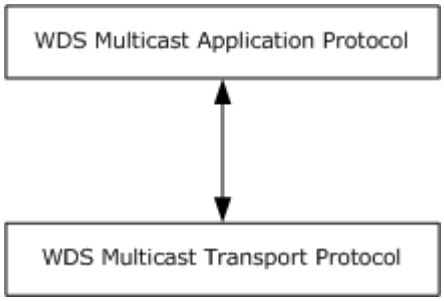


Figure 1: Relationship between Multicast Application Protocol and Multicast Transport Protocol

1.5 Prerequisites/Preconditions

The protocol relies on WDS Multicast Session Initiation Protocol to provide each client with the details of the content and parameters required for WDS Multicast Transport Protocol.

The protocol assumes that WDS Multicast Session Initiation Protocol has created and initialized instances of WDS Multicast Application Protocol and WDS Multicast Transport Protocol.

Copyright © 2013 Microsoft Corporation.

1.6 Applicability Statement

WDS Multicast Application Protocol is applicable when an application needs to download content from a server using a multicast session.

1.7 Versioning and Capability Negotiation

This protocol does not have any explicit versioning negotiation.

1.8 Vendor-Extensible Fields

This protocol does not have any vendor-extensible fields.

1.9 Standards Assignments

None.

Copyright © 2013 Microsoft Corporation.

2 Messages

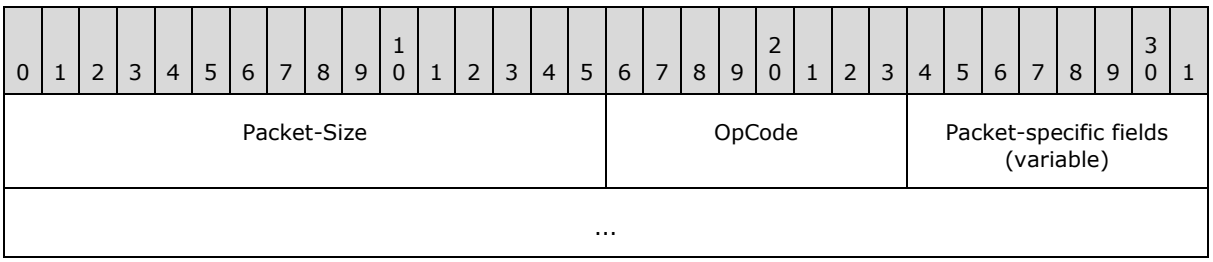
2.1 Transport

WDS Multicast Application Protocol MUST be network-byte-order unless noted otherwise.

2.2 Message Syntax

2.2.1 Packet Header

Each packet of the WDS Multicast Application Protocol MUST specify the packet header as follows:



Packet-Size (2 bytes): Specifies the total length of the packet in bytes.

OpCode (1 byte): A unique numeric value that is assigned to each packet of WDS Multicast Application Protocol and is used to identify the format for the rest of the packet.

Packet-specific fields (variable): Based on the **OpCode** field of the packet header, additional fields may be specified for each **OpCode** as specified below:

The following table specifies the possible values for the **OpCode** field:

OpCode	Meaning
WDSMC_PKTYPE_SRV_CIR 0x01	Section 2.2.2
WDSMC_PKTYPE_CNT_CIR 0x02	Section 2.2.3
WDSMC_PKTYPE_SRV_DATA 0x03	Section 2.2.4
WDSMC_PKTYPE_CNT_PROGRESS 0x04	Section 2.2.5

2.2.2 SRVCIR Packet

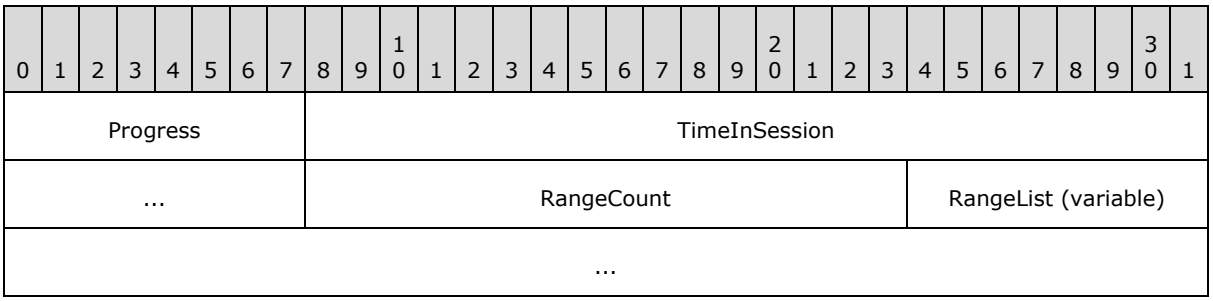
This packet is sent by the server to all clients using the **POLL Trigger** provided by the WDS Multicast Transport Protocol.

This packet does not specify any additional fields except those specified for the packet header (section 2.2.1).

Copyright © 2013 Microsoft Corporation.

2.2.3 CNTCIR Packet

This packet is sent by the client in response to the SRVCIR Packet (section 2.2.2) and is delivered to the server by the **POLLACK Trigger** from the WDS Multicast Transport Protocol.



Progress (1 byte): MUST be set to a numeric value ranging from 0–100 specifying the percentage of the blocks of content that have been received by client.

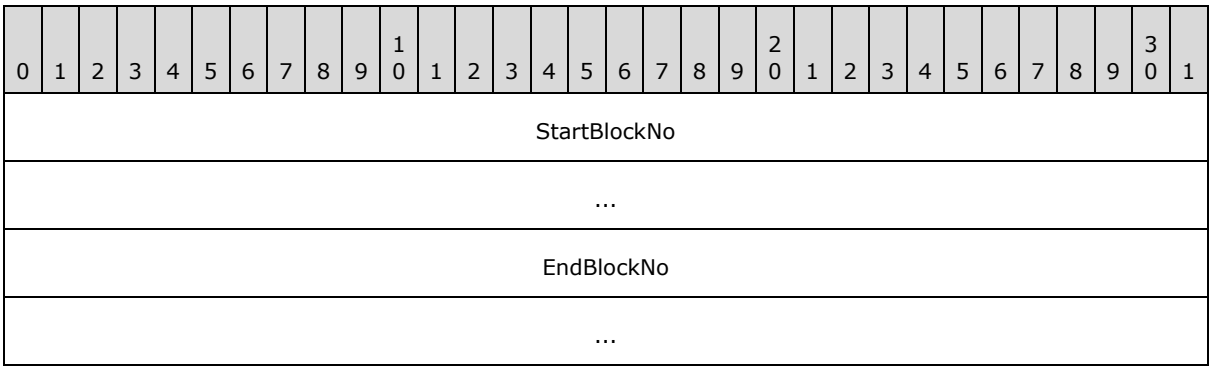
TimeInSession (4 bytes): MUST be set to the number of seconds elapsed since the client joined the multicast session.

RangeCount (2 bytes): MUST be set to the number of block ranges of the content that the client is currently missing and that are specified in the **RangeList** field. The maximum number of ranges MUST NOT exceed 64. If the number of missing ranges on the client exceeds 64, then the client MUST send only the first 64 block ranges.

RangeList (variable): MUST be set to the block ranges of content that the client is missing as specified in section 2.2.3.1. The count of ranges specified MUST match the count specified by the **RangeCount** field.

2.2.3.1 Range List

This field specifies an array of block ranges of content that the client is missing. Each element of the array MUST specify the range as follows:

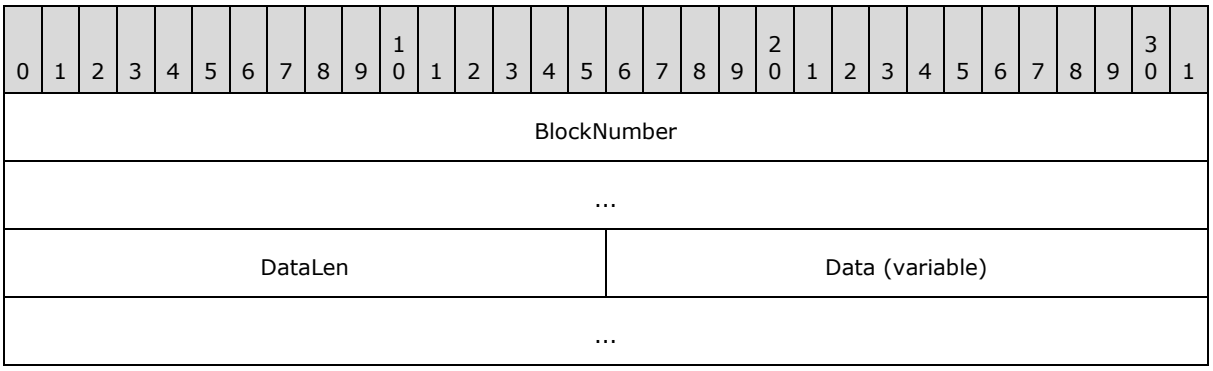


StartBlockNo (8 bytes): MUST be set to the starting BlockNumber of the range of **block numbers** missing for the content.

EndBlockNo (8 bytes): MUST be set to the ending BlockNumber of the range of block numbers missing for the content.

Copyright © 2013 Microsoft Corporation.

2.2.4 DATA Packet

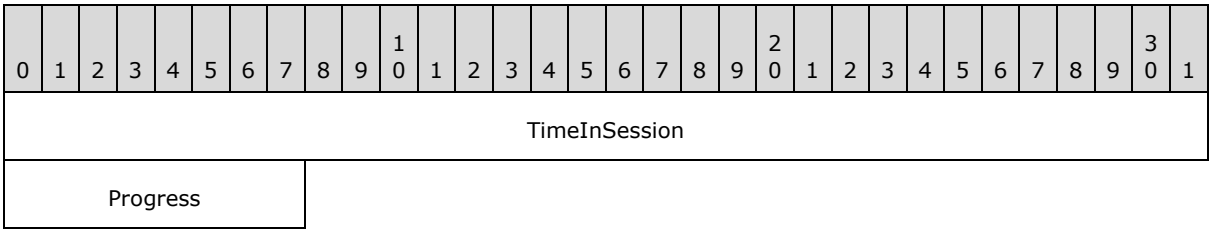


BlockNumber (8 bytes): MUST be set to the unique block number for the data being sent for the content.

DataLen (2 bytes): MUST be set to the length in bytes of the payload included in the **Data** field.

Data (variable): MUST be set to the data read from the content.

2.2.5 PROGRESS Packet



TimeInSession (4 bytes): MUST be set to the number of seconds elapsed since the client joined the multicast session.

Progress (1 byte): MUST be set to a numeric value ranging from 0-100 specify the percentage of the Blocks of Content that have been received by client.

Copyright © 2013 Microsoft Corporation.

3 Protocol Details

3.1 Server Details

This section specifies the WDS Multicast Application Protocol behavior for the server.

The following state diagram shows the lifetime of the protocol on the server:

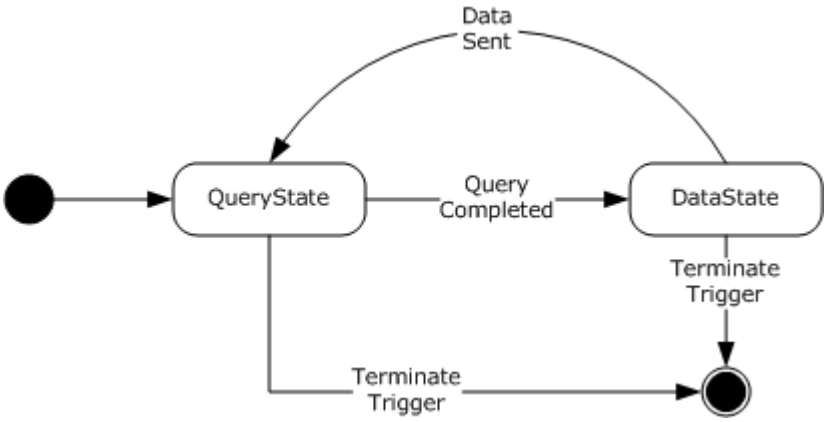


Figure 2: Server state diagram

3.1.1 Abstract Data Model

This section describes a conceptual model of possible data organization that an implementation maintains to participate in this protocol. The described organization is provided to facilitate the explanation of how the protocol behaves. This document does not mandate that implementations adhere to this model as long as their external behavior is consistent with that described in this document.

SessionState: Specifies the current state for the multicast session as defined below.

State	Description
QueryState	WDS Multicast Application Protocol is actively querying the clients for block ranges of content each client is missing.
DataState	WDS Multicast Application Protocol is sending the data for missing block ranges discovered in QueryState.

MUST be set to QueryState on initialization.

BlockSize: Specifies the block size to use to send data to clients. The value is provided by WDS Multicast Session Initiation Protocol.

MissingBlockRanges: Specifies an array of missing block ranges that MUST be constructed from the CNTCIR packets received from clients as specified in section 3.1.6.1. Each element of the array has two elements as specified below:

StartBlockNo: Specifies the starting block number for the missing block range.

EndBlockNo: Specifies the ending block number for the missing block range.

Copyright © 2013 Microsoft Corporation.

ClientCNTCIRPackets: Specifies the list of CNTCIR packets (section 2.2.3) received from clients while server is in QueryState.

3.1.2 Timers

Timer	Description
Query Timer	The timeout for this timer is provided by the WDS Multicast Transport Protocol in response to **POLL Trigger**.

3.1.3 Initialization

Server MUST be initialized for **QueryState** and MUST start processing as specified in section 3.1.5.1.

3.1.4 Higher-Layer Triggered Events

None.

3.1.5 Message Processing Events and Sequencing Rules

WDS Multicast Application Protocol drives the processing based on Lower-Layered Triggered Events generated by WDS Multicast Transport Protocol.

The following table specifies the Lower-Layered Triggered Events generated by WDS Multicast Transport Protocol and specifies the processing for each:

Lower-Layered Triggered Events	Description
POLLACK Trigger	Section 3.1.5.1.1
Status Trigger	Section 3.1.5.3
Data Empty Trigger	Section 3.1.5.2.1
Terminate Trigger	Section 3.1.5.4

3.1.5.1 Query State

Server MUST remove all packets stored in **ClientCNTCIRPackets** (section 3.1.1).

Server MUST construct a SRVCIR packet (section 2.2.2) and MUST send the packet using **POLL Trigger** to the WDS Multicast Transport Protocol. The timeout value provided by **POLL Trigger** MUST be used to set the expiry time for the **Query Timer**.

3.1.5.1.1 POLLACK Trigger

POLLACK Trigger MUST provide the payload received from the client. Server MUST validate that the payload specifies a CNTCIR packet as specified in section 2.2.3 and MUST add the packet to **ClientCNTCIRPackets** (section 3.1.1).

3.1.5.2 Data State

The server MUST start from the first block number of the first block range specified by **MissingBlockRanges** (section 3.1.1) and MUST process for each block number as specified below:

Copyright © 2013 Microsoft Corporation.

1. The server MUST compute the offset into the content as follows:

   BlockOffset = **BlockNumber** x **BlockSize** (section 3.1.1)

2. The server MUST read the bytes of data from the content at offset specified by **BlockOffset**. The number of bytes read MUST be equal to the **BlockSize** (section 3.1.1), unless the read operation reaches to the last byte of the content in which case there will be fewer bytes read.

3. The server MUST construct a DATA packet and set the fields of the DATA packet as specified below:

   BlockNumber: MUST be set to the **BlockNumber**.

   DataLen: MUST be set to the number of bytes of data read from the content.

   Data: MUST be set the bytes of data read from the content.

4. The server MUST send the **Data Trigger** to the WDS Multicast Transport Protocol providing the constructed DATA packet.

Server MUST wait for the **Data Empty Trigger** from the WDS Multicast Transport Protocol.

3.1.5.2.1 Data Empty Trigger

Server MUST ignore **Data Empty Trigger** when **SessionState** (section 3.1.1) is not set to **DataState**.

Otherwise, server MUST change the **SessionState** (section 3.1.1) to **QueryState** and MUST continue processing as specified in section 3.1.5.1.

3.1.5.3 Status Trigger

Status Trigger MUST provide the payload received from client. Server MUST validate that the payload specifies a PROGRESS packet (section 2.2.5).

3.1.5.4 Terminate Trigger

When Terminate Trigger is received, server MUST shut down the WDS Multicast Application Protocol.

3.1.6 Timer Events

3.1.6.1 Query Timer

If the **ClientCNTCIRPackets** (section 3.1.1) is empty, the server MUST continue processing as specified in section 3.1.5.1.

If the **ClientCNTCIRPackets** is not empty, the server MUST remove all entries stored in **MissingBlockRanges** (section 3.1.1). The server MUST merge block ranges specified by CNTCIR packets stored in **ClientCNTCIRPacket** and store the merged block ranges list into **MissingBlockRanges**.

To merge the block ranges, the server MUST first eliminate the CNTCIR packet (section 2.2.3) as follows:

1. Find the CNTCIR packet which has the highest value for the **TimeInSession** field (section 2.2.3).

Copyright © 2013 Microsoft Corporation.

2. Delete all such CNTCIR packets in the **ClientCNTCIRPackets** (section where the **TimeInSession** field specifies that the client joined after 30 seconds of the oldest client.

Server MUST construct a merged list of missing block ranges from the remaining CNTCIR packets in **ClientCNTCIRPackets** and set the **MissingBlockRanges** to the merged list.

The merged list MUST NOT have any overlapping block ranges and MUST be in ascending order.

Server MUST change the **SessionState** (section 3.1.1) to **DataState** and MUST disable **Query Timer** and continue processing as specified in section 3.1.5.2.

3.1.7 Other Local Events

None.

3.2 Client Details

3.2.1 Abstract Data Model

This section describes a conceptual model of possible data organization that an implementation maintains to participate in this protocol. The described organization is provided to facilitate the explanation of how the protocol behaves. This document does not mandate that implementations adhere to this model as long as their external behavior is consistent with that described in this document.

TotalBlocks: Specifies the number of blocks in the content. This value is provided by WDS Multicast Session Initiation Protocol.

MissingBlocksBitmap: An array of bits with the size of the array set to equal **TotalBlocks**. Each bit is initialized to zero, and when a DATA packet is received for the specified block number, the bit at that index is changed to one. The array uses 1-based index.

BlockSize: Specifies the number of bytes of data from the content sent in each DATA packet. This value is provided by WDS Multicast Session Initiation Protocol.

MaxCacheSize: Specifies the maximum number of bytes of DATA packets that can be processed by client before the WDS Multicast Transport Protocol stops sending the **DATA Trigger**, and MUST wait for a **Cache Done Trigger** from WDS Multicast Application Protocol. The default is 2097152.

JoinTime: MUST be set to local time in second granularity on initialization.

3.2.2 Timers

None.

3.2.3 Initialization

On initialization client MUST wait for packets to arrive from the server using the Lower-Layered Triggered Events as specified in section 3.2.5.

3.2.4 Higher-Layer Triggered Events

None.

Copyright © 2013 Microsoft Corporation.

3.2.5 Message Processing Events and Sequencing Rules

The client processes the Lower-Layered Triggered Events from the WDS Multicast Transport Protocol, as specified in the following sections.

3.2.5.1 DATA Trigger

The client MUST validate that the payload provided by the DATA Trigger specifies the DATA packet as per section 2.2.4. The client MUST check the value of the bit at the index specified by the **BlockNumber** field of the DATA packet in **MissingBlocksBitmap** (section 3.2.1). If the value is set to one, then the client MUST ignore the DATA packet and stop further processing.

If the bit value is not set to one, the client MUST set the bit to one and MUST send the **Cache Done Trigger** to the WDS Multicast Transport Protocol specifying the size of the DATA packet.

The client MUST check whether all bits of **MissingBlocksBitmap** (section 3.2.1) are now set to one. If so then the download of Content has completed and the client MUST terminate.

3.2.5.2 Query Cache Trigger

The client MUST reply to the trigger by providing the **MaxCacheSize** value (section 3.2.1).

3.2.5.3 QCC Trigger

When the **QCC Trigger** is received, the client MUST construct the PROGRESS packet (section 2.2.5) and set the fields of the packet as specified below:

TimeInSession: MUST be set to (Local Time in Seconds – **JoinTime**) (section 3.2.1).

Progress: MUST be set to the percentage of bits that are set to zero in MissingBlocksBitmap (section 3.2.1).

The client MUST provide the constructed packet in reply to the **QCC Trigger**.

3.2.5.4 POLL Trigger

When a POLL Trigger is received, the client MUST construct a CNTCIR packet (section 2.2.3) and set the fields of the packet as follows:

TimeInSession: MUST be set to (Local Time in Seconds – **JoinTime** (section 3.2.1)).

Progress: MUST be set to the percentage of bits that are set to zero in **MissingBlocksBitmap** (section 3.2.1) and MUST be in range 0-100.

RangeList: The server MUST go through the **MissingBlocksBitmap** (section 3.2.1) to find the ranges of blocks that are missing (such blocks will have the respective bit set to zero) and MUST add all such missing ranges to the **RangeList** field. The number of ranges added MUST NOT exceed the limit specified in section 2.2.3.1.

RangeCount: MUST be set to the ranges specified by the **RangeList**.

Client MUST provide the constructed packet in reply to **POLL Trigger**.

3.2.6 Timer Events

None.

Copyright © 2013 Microsoft Corporation.

3.2.7 Other Local Events

None.

Copyright © 2013 Microsoft Corporation.

4 Protocol Examples

The following diagram specifies a server querying 3 clients and sending DATA packets:

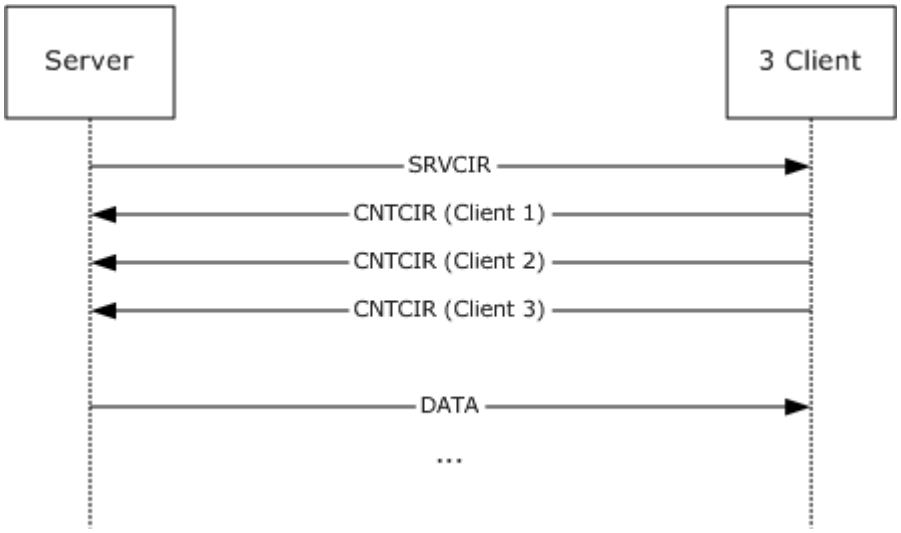


Figure 3: Client-server interaction

Copyright © 2013 Microsoft Corporation.

5 Security

5.1 Security Considerations for Implementers

WDS Multicast Application Protocol relies on the WDS Multicast Transport Protocol to provide security.

5.2 Index of Security Parameters

None.

Copyright © 2013 Microsoft Corporation.

6 Appendix A: Product Behavior

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include released service packs:

- Windows Vista operating system
- Windows Server 2008 operating system
- Windows 7 operating system
- Windows Server 2008 R2 operating system
- Windows 8 operating system
- Windows Server 2012 operating system
- Windows 8.1 operating system
- Windows Server 2012 R2 operating system

Exceptions, if any, are noted below. If a service pack or Quick Fix Engineering (QFE) number appears with the product version, behavior changed in that service pack or QFE. The new behavior also applies to subsequent service packs of the product unless otherwise specified. If a product edition appears with the product version, behavior is different in that product edition.

Unless otherwise specified, any statement of optional behavior in this specification that is prescribed using the terms SHOULD or SHOULD NOT implies product behavior in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term MAY implies that the product does not follow the prescription.

Copyright © 2013 Microsoft Corporation.

7 Change Tracking

This section identifies changes that were made to the [MS-WDSMA] protocol document between the January 2013 and August 2013 releases. Changes are classified as New, Major, Minor, Editorial, or No change.

The revision class **New** means that a new document is being released.

The revision class **Major** means that the technical content in the document was significantly revised. Major changes affect protocol interoperability or implementation. Examples of major changes are:

- A document revision that incorporates changes to interoperability requirements or functionality.
- An extensive rewrite, addition, or deletion of major portions of content.
- The removal of a document from the documentation set.
- Changes made for template compliance.

The revision class **Minor** means that the meaning of the technical content was clarified. Minor changes do not affect protocol interoperability or implementation. Examples of minor changes are updates to clarify ambiguity at the sentence, paragraph, or table level.

The revision class **Editorial** means that the language and formatting in the technical content was changed. Editorial changes apply to grammatical, formatting, and style issues.

The revision class **No change** means that no new technical or language changes were introduced. The technical content of the document is identical to the last released version, but minor editorial and formatting changes, as well as updates to the header and footer information, and to the revision summary, may have been made.

Major and minor changes can be described further using the following change types:

- New content added.
- Content updated.
- Content removed.
- New product behavior note added.
- Product behavior note updated.
- Product behavior note removed.
- New protocol syntax added.
- Protocol syntax updated.
- Protocol syntax removed.
- New content added due to protocol revision.
- Content updated due to protocol revision.
- Content removed due to protocol revision.
- New protocol syntax added due to protocol revision.

Copyright © 2013 Microsoft Corporation.

- Protocol syntax updated due to protocol revision.
- Protocol syntax removed due to protocol revision.
- New content added for template compliance.
- Content updated for template compliance.
- Content removed for template compliance.
- Obsolete document removed.

Editorial changes are always classified with the change type **Editorially updated.**

Some important terms used in the change type descriptions are defined as follows:

- **Protocol syntax** refers to data elements (such as packets, structures, enumerations, and methods) as well as interfaces.
- **Protocol revision** refers to changes made to a protocol that affect the bits that are sent over the wire.

The changes made to this document are listed in the following table. For more information, please contact protocol@microsoft.com.

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
6 Appendix A: Product Behavior	Modified this section to include references to Windows 8.1 operating system and Windows Server 2012 R2 operating system.	Y	Content updated.

Copyright © 2013 Microsoft Corporation.

8 Index

Copyright © 2013 Microsoft Corporation.

Copyright © 2013 Microsoft Corporation.