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.

- **Fictitious Names.** The example companies, organizations, products, domain names, e-mail 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.
# Revision Summary

<table>
<thead>
<tr>
<th>Date</th>
<th>Revision History</th>
<th>Revision Class</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>07/13/2009</td>
<td>0.1</td>
<td>Major</td>
<td>Initial Availability</td>
</tr>
<tr>
<td>08/28/2009</td>
<td>0.2</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>11/06/2009</td>
<td>0.3</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>02/19/2010</td>
<td>1.0</td>
<td>Minor</td>
<td>Updated the technical content</td>
</tr>
<tr>
<td>03/31/2010</td>
<td>1.01</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>04/30/2010</td>
<td>1.02</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>06/07/2010</td>
<td>1.03</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>06/29/2010</td>
<td>1.04</td>
<td>Minor</td>
<td>Clarified the meaning of the technical content.</td>
</tr>
<tr>
<td>07/23/2010</td>
<td>1.04</td>
<td>No change</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>09/27/2010</td>
<td>1.04</td>
<td>No change</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>11/15/2010</td>
<td>1.04</td>
<td>No change</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>12/17/2010</td>
<td>1.04</td>
<td>No change</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>03/18/2011</td>
<td>1.04</td>
<td>No change</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>06/10/2011</td>
<td>1.04</td>
<td>No change</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>01/20/2012</td>
<td>1.5</td>
<td>Minor</td>
<td>Clarified the meaning of the technical content.</td>
</tr>
<tr>
<td>04/11/2012</td>
<td>1.5</td>
<td>No change</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>07/16/2012</td>
<td>1.5</td>
<td>No change</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
</tbody>
</table>
# Table of Contents

1 Introduction .......................................................................................................................... 7
   1.1 Glossary ......................................................................................................................... 7
   1.2 References ...................................................................................................................... 7
      1.2.1 Normative References ............................................................................................... 8
      1.2.2 Informative References ............................................................................................ 8
   1.3 Protocol Overview (Synopsis) ........................................................................................ 8
   1.4 Relationship to Other Protocols .................................................................................... 10
   1.5 Prerequisites/Preconditions ............................................................................................ 11
   1.6 Applicability Statement .................................................................................................. 11
   1.7 Versioning and Capability Negotiation ............................................................................ 11
   1.8 Vendor-Extensible Fields ............................................................................................... 11
   1.9 Standards Assignments ................................................................................................... 11

2 Messages .............................................................................................................................. 12
   2.1 Transport ......................................................................................................................... 12
   2.2 Common Data Types ....................................................................................................... 12
      2.2.1 Simple Data Types and Enumerations ...................................................................... 12
         2.2.1.1 Task Type ........................................................................................................... 12
         2.2.1.2 Catalog ID .......................................................................................................... 12
         2.2.1.3 Propagation Error Type ..................................................................................... 12
      2.2.2 Bit Fields and Flag Structures .................................................................................. 13
      2.2.3 Binary Structures ..................................................................................................... 13
         2.2.3.1 Full-Text Index Component Message .................................................................. 13
            2.2.3.1.1 Propagation List File .................................................................................... 13
               2.2.3.1.1.1 String Record ....................................................................................... 13
               2.2.3.1.1.2 String List ............................................................................................ 13
            2.2.3.2 Versioned Index Identifier .............................................................................. 14
      2.2.4 Result Sets ................................................................................................................ 14
      2.2.5 Tables and Views ...................................................................................................... 14
      2.2.6 XML Structures ....................................................................................................... 14
         2.2.6.1 Namespaces ......................................................................................................... 14
         2.2.6.2 Simple Types ....................................................................................................... 14
         2.2.6.3 Complex Types ................................................................................................... 14
         2.2.6.4 Elements ............................................................................................................ 14
         2.2.6.5 Attributes ........................................................................................................... 15
         2.2.6.6 Groups ................................................................................................................ 15
         2.2.6.7 Attribute Groups ............................................................................................... 15

3 Protocol Details .................................................................................................................... 16
   3.1 Back-End Database Server Details ................................................................................. 16
      3.1.1 Abstract Data Model ................................................................................................ 16
         3.1.1.1 List of Ready Query Components ...................................................................... 16
         3.1.1.2 List of Running Tasks ....................................................................................... 16
         3.1.1.3 List of Propagation Errors ................................................................................. 17
      3.1.2 Timers ....................................................................................................................... 17
      3.1.3 Initialization .............................................................................................................. 17
      3.1.4 Higher-Layer Triggered Events ................................................................................. 18
      3.1.5 Message Processing Events and Sequencing Rules .................................................. 18
         3.1.5.1 proc_MSS_PropagationIndexerCleanUpTablesForTask ...................................... 18
         3.1.5.2 proc_MSS_PropagationIndexerGetCompletedTasks ........................................... 19
3.1.5.2.1 Completed Tasks Result Set ............................................... 19
3.1.5.3 proc_MSS_PropagationIndexerGetReadyQueryComponents ................. 20
3.1.5.3.1 Ready Query Components Result Set ..................................... 20
3.1.5.4 proc_MSS_PropagationIndexerGetTasks .................................... 20
3.1.5.4.1 Propagation Tasks Result Set ........................................... 21
3.1.5.5 proc_MSS_PropagationIndexerInsertNewTask ................................ 21
3.1.5.6 proc_MSS_PropagationQueryComponentPickUpNewPropagationItems ....... 22
3.1.5.6.1 Propagation Tasks Result Set ........................................... 23
3.1.5.7 proc_MSS_PropagationIndexerDeleteAllTasksFromSender .................. 24
3.1.5.8 proc_MSS_PropagationQueryComponentReportTaskReady ................... 24
3.1.5.9 proc_MSS_PropagationReportError ........................................ 26
3.1.5.10 proc_MSS_PropagationDeleteError ....................................... 26
3.1.5.11 proc_MSS_PropagationDeleteErrors .................................... 27
3.1.5.12 proc_MSS_PropagationGetErrors ........................................ 28
3.1.5.12.1 Propagation Errors Result Set ....................................... 28
3.1.5.13 proc_MSS_PropagationGetTasks ......................................... 28
3.1.5.13.1 Propagation All Tasks Result Set ................................... 29
3.1.5.14 proc_MSS_PropagationGetTaskCompletions ................................ 29
3.1.5.14.1 Propagation Task Completions Result Set .............................. 29
3.1.6 Timer Events ........................................................................ 30
3.1.7 Other Local Events .................................................................. 30
3.2 Sender Details ........................................................................... 30
  3.2.1 Abstract Data Model ................................................................. 30
  3.2.1.1 Search Application Name ..................................................... 30
  3.2.1.2 Sender Identification ......................................................... 30
  3.2.1.3 List of Ready Query Components ......................................... 30
  3.2.1.4 List of Running Tasks ........................................................ 31
  3.2.1.5 List of Completed Tasks .................................................... 31
  3.2.1.6 Error Possibly Exists ....................................................... 31
  3.2.2 Timers .................................................................................. 31
  3.2.3 Initialization .......................................................................... 31
  3.2.4 Higher-Layer Triggered Events ................................................. 31
  3.2.5 Message Processing Events and Sequencing Rules ........................... 32
    3.2.5.1 Sending a proc_MSS_PropagationIndexerGetReadyQueryComponents Message .......................................................... 32
    3.2.5.2 Receiving a Ready Query Components Result Set ....................... 32
    3.2.5.3 Sending a Full-Text Index Component Message ............................. 32
    3.2.5.4 Sending a proc_MSS_PropagationReportError Message .................. 33
    3.2.5.5 Sending a proc_MSS_PropagationDeleteError Message .................. 34
    3.2.5.6 Sending a proc_MSS_PropagationIndexerInsertNewTask Message ........ 34
    3.2.5.7 Sending a proc_MSS_PropagationIndexerGetCompletedTasks Message .... 35
    3.2.5.8 Receiving a Completed Tasks Result Set Message ........................ 36
    3.2.5.9 Sending a proc_MSS_PropagationIndexerCleanUpTablesForTask Message ...... 36
    3.2.5.10 Sending the proc_MSS_PropagationIndexerGetTasks Message .......... 36
    3.2.5.11 Receiving a Propagation Tasks Result Set ................................ 37
    3.2.5.12 Sending the proc_MSS_PropagationIndexerDeleteAllTasksFromSender Message .......................................................... 37
    3.2.6 Timer Events ...................................................................... 37
    3.2.7 Other Local Events ............................................................... 37
3.3 Receiver Details .......................................................................... 37
  3.3.1 Abstract Data Model ................................................................. 37
    3.3.1.1 Receiver Identifier ............................................................. 37
    3.3.1.2 List of Incomplete Tasks ..................................................... 38
3.3.1.3 Error Possibly Exists .................................................. 38
3.3.2 Timers ........................................................................ 38
3.3.3 Initialization .............................................................. 38
3.3.4 Higher-Layer Triggered Events ................................. 38
3.3.5 Message Processing Events and Sequencing Rules .... 38
    3.3.5.1 Sending a proc_MSS_PropagationQueryComponentPickUpNewPropagationItems Message .................................................. 38
    3.3.5.2 Receiving a Propagation Tasks Result Set ...................... 39
    3.3.5.3 Sending a proc_MSS_PropagationReportError Message .......... 39
    3.3.5.4 Sending a proc_MSS_PropagationDeleteError Message .......... 40
    3.3.5.5 Sending a proc_MSS_PropagationQueryComponentReportTaskReady Message .................................................. 40
3.3.6 Timer Events ............................................................... 41
3.3.7 Other Local Events ..................................................... 41
3.4 Admin Server Details ...................................................... 41
    3.4.1 Abstract Data Model ................................................... 41
    3.4.1.1 List of Running Tasks ............................................... 41
    3.4.1.2 List of Task Completions ........................................... 41
    3.4.1.3 List of Propagation Errors .......................................... 42
    3.4.1.4 List of Receivers .................................................... 42
    3.4.2 Timers ...................................................................... 42
    3.4.3 Initialization ............................................................. 42
    3.4.4 Higher-Layer Triggered Events ...................................... 43
    3.4.5 Message Processing Events and Sequencing Rules .... 43
        3.4.5.1 Sending a proc_MSS_PropagationGetTasks Message ............... 43
        3.4.5.2 Receiving a Propagation All Tasks Result Set ...................... 43
        3.4.5.3 Sending a proc_MSS_PropagationGetTaskCompletions Message .......... 43
        3.4.5.4 Receiving a Propagation Task Completions Result Set .......... 43
        3.4.5.5 Sending a proc_MSS_PropagationGetErrors Message .......... 44
        3.4.5.6 Receiving a Propagation Errors Result Set ....................... 44
        3.4.5.7 Sending a proc_MSS_GetQueryComponents Message .............. 45
        3.4.5.8 Receiving a Query Components Result Set Message .......... 45
        3.4.5.9 Sending a proc_MSS_GetComponentStatusUpToDate Message .... 45
        3.4.5.10 Sending a proc_MSS_PropagationReportError Message ......... 46
        3.4.5.11 Sending a proc_MSS_PropagationDeleteError Message .......... 47
        3.4.5.12 Sending a proc_MSS_SetQueryComponent Message .............. 47
3.4.6 Timer Events ............................................................... 47
3.4.7 Other Local Events ..................................................... 47

4 Protocol Examples .......................................................... 48
4.1 Component Addition Propagation ...................................... 48
    4.1.1 Initial State ............................................................. 48
        4.1.1.1 DB-1 .................................................................. 48
            4.1.1.1.1 List of Ready Query Components ......................... 48
            4.1.1.1.2 List of Running Tasks ...................................... 48
        4.1.1.2 SEN-1 .............................................................. 48
            4.1.1.2.1 Search Application Name .................................. 48
            4.1.1.2.2 Sender Identifier ............................................. 48
            4.1.1.2.3 List of Ready Query Components ......................... 49
            4.1.1.2.4 List of Completed Tasks .................................... 49
        4.1.1.3 REC-1 .............................................................. 49
            4.1.1.3.1 Receiver Identifier .......................................... 49

[MS-CIPROP2] — v20120630
Index Propagation Version 2 Protocol Specification

Copyright © 2012 Microsoft Corporation.

Release: July 16, 2012
4.1.1.3.2 List of Incomplete Tasks .......................................................... 49
4.1.1.4 REC-2 .......................................................................................... 49
4.1.1.4.1 Receiver Identifier ................................................................. 49
4.1.1.4.2 List of Incomplete Tasks ....................................................... 49
4.1.2 Sequence of Events ...................................................................... 49

5 Security .................................................................................................. 53
5.1 Security Considerations for Implementers ........................................ 53
5.2 Index of Security Parameters .......................................................... 53

6 Appendix A: Product Behavior ............................................................ 54

7 Change Tracking .................................................................................. 55

8 Index ...................................................................................................... 56
Introduction

This document specifies the Index Propagation Version 2 Protocol. It is a complete protocol, not an extension of an existing one. The protocol is used to replicate search index data across multiple servers and to maintain consistency among those servers in the event of changes to that data.

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 defined in [MS-GLOS]:

- Coordinated Universal Time (UTC)
- little-endian
- Security Support Provider Interface (SSPI)
- Unicode

The following terms are defined in [MS-OFCGLOS]:

- back-end database server
- component birth date
- crawl component
- datetime
- document identifier
- document set
- farm
- full-text index catalog
- full-text index component
- group
- index identifier
- query component
- query topology
- result set
- return code
- search application
- search service application
- security group
- static rank
- stored procedure
- Transact-Structured Query Language (T-SQL)

The following terms are specific to this document:

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 technical documents, which are updated frequently. References to other documents include a publishing year when one is available.
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.


[MS-CIFO] Microsoft Corporation, "Content Index Format Structure Specification".


1.2.2 Informative References


[MS-OFCGLOS] Microsoft Corporation, "Microsoft Office Master Glossary".

1.3 Protocol Overview (Synopsis)

This document specifies communication between a crawl component (the sender) and a query component (2) (the receiver). This protocol only applies to the activity of replicating full-text index catalog data from the sender into the full-text index catalog data used by the receiver serving a search service application.

This protocol is used to synchronize changes made to a full-text index catalog from either static rank computation or addition of a full-text index component across receivers. The process of sending messages to receivers to ensure that the same index operation is applied to all replicated full-text index catalogs is called "propagation". These operations are referred to in this document as "propagation tasks". One of these tasks, component addition, includes additions, revisions, and removals of crawled content. Propagation of the static rank computation task is also necessary, because equivalent queries may be routed to different receivers on successive requests, and performing static rank computation on all receivers ensures retrieval of the same results across multiple requests.

Senders inform the back-end database server of any changes, while receivers regularly poll back-end database server for timely propagation of changes and updates. On the other hand, receivers inform the back-end database server that they are up-to-date and this information is propagated to senders through the back-end database server. Also, senders transport the full-text index catalog to receivers using the SMB protocol; the format of these catalogs is described in [MS-CIFO].
The admin server periodically interprets the current list of propagation errors and takes query components (2) offline when appropriate.

This protocol specification applies independently to each search application. If there are two or more search applications on a farm, they will all have same requirements for the implementation of this protocol and will be independent of each other.

The following figure shows a high-level view of the propagation process and the stored procedures involved.

![High-level view of communication between servers](image)

**Figure 1: High-level view of communication between servers**

The following figure shows the sequence of events during a particular propagation instance.
Figure 2: Sequence of operations used to propagate the full-text index component

1.4 Relationship to Other Protocols

The Tabular Data Stream protocol, as described in [MS-TDS], is the transport protocol used to call the stored procedures, query SQL views or SQL tables, return result sets and return codes.

This protocol relies on Server Message Control Block (SMB), as described in [MS-SMB], as its transport protocol to perform server-to-server copies of full-text index catalogs.
1.5 Prerequisites/Preconditions

This protocol requires that a farm be installed and configured. The operations described by the protocol operate between a client that is a part of the farm and a back-end database server on which the databases of the farm are stored.

The user that calls the stored procedures specified in this document has permission to read from and write to the databases that contain those stored procedures.

The following prerequisites are also required before the propagation protocol can be successfully invoked. This protocol assumes that the following conditions are true:

- There is a file system share on each query server that allows read and write operation by the local security group named "WSS_WPG" on that query server.
- The stored procedures specified in this document are present on the back-end database server.
- The servers on which the sender and receiver run are members of the farm.

1.6 Applicability Statement

This protocol is applicable only to the activity of replicating full-text index catalog data from a crawl component into the full-text index catalog data used by query components (2) serving one particular search service application. The protocol is designed for use by no more than 10 senders and 20 receivers, propagating no more than 5 full-text index components per second per sender.

1.7 Versioning and Capability Negotiation

This document covers versioning issues in the following areas:

- **Supported Transports:** This protocol uses the SMB protocol, as described in [MS-SMB], for file copies, and the TDS protocol, as described in [MS-TDS], for SQL stored procedure calls.

- **Security and Authentication Methods:** This protocol supports the Security Support Provider Interface (SSPI) and SQL authentication with the protocol server role described in [MS-TDS].

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.
2 Messages

2.1 Transport

The TDS protocol, as specified in [MS-TDS], is the transport protocol used to call the stored procedures, query SQL views or SQL tables, return codes, and return result sets.

The SMB protocol, as specified in [MS-SMB], is the transport protocol used to copy files to another server.

2.2 Common Data Types

2.2.1 Simple Data Types and Enumerations

2.2.1.1 Task Type

A 32-bit signed integer used to represent the type of a propagation task. It MUST be one of the values in the following table.

<table>
<thead>
<tr>
<th>Symbolic name</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ComponentAddition</td>
<td>1</td>
<td>A full-text index component is received by each receiver.</td>
</tr>
<tr>
<td>StaticRankComputation</td>
<td>2</td>
<td>All activities performed during a static rank computation event are performed by each receiver.</td>
</tr>
</tbody>
</table>

2.2.1.2 Catalog ID

A 32-bit signed integer used to represent a full-text index catalog. It MUST be one of the values in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The main catalog, as specified in [MS-CIFO] section 2.18.1.</td>
</tr>
<tr>
<td>2</td>
<td>The anchor text catalog, as specified in [MS-CIFO] section 2.18.2.</td>
</tr>
</tbody>
</table>

2.2.1.3 Propagation Error Type

A 32-bit signed integer used to represent common categories of error encountered by participants in this protocol. It MUST be one of the values in the following table.

<table>
<thead>
<tr>
<th>Symbolic name</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FileCopy</td>
<td>0</td>
<td>A sender could not copy a full-text index component to a receiver.</td>
</tr>
<tr>
<td>IndexAbsorption</td>
<td>1</td>
<td>A receiver encountered an error while processing a component addition propagation task.</td>
</tr>
<tr>
<td>IndexCorruption</td>
<td>2</td>
<td>The full-text index catalog on a receiver was found to contain incorrect data.</td>
</tr>
<tr>
<td>ReceiverHang</td>
<td>3</td>
<td>At least one propagation task was not finished by a receiver in the expected time.</td>
</tr>
</tbody>
</table>
2.2.2 Bit Fields and Flag Structures

None.

2.2.3 Binary Structures

2.2.3.1 Full-Text Index Component Message

The unit of transfer in full-text index component propagation is a set of files. Each file in this set, except for one, is a duplicate of a file in a full-text index component, as specified in [MS-CIFO] section 2.17, in content, but the extension ".cp" is appended to the original file name to create the name of the duplicate file. Every file of a full-text index component is represented in the set.

The one file in this set that does not correspond to a full-text index component file is a propagation list file, as specified in section 2.2.3.1.1. See section 4.1 for an example.

2.2.3.1.1 Propagation List File

The ".list" file is a list of Unicode strings stored in the string list format specified in section 2.2.3.1.1.2. All integers and characters are stored in little-endian form unless specified otherwise.

2.2.3.1.1.1 String Record

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 1 |
|   |   |   |   |   | 1 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 1 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 3 | 0 | 1 |

[previous record]  Number of Characters

...  Characters (variable)

**Number of Characters (4 bytes):** A 32-bit unsigned integer representing the number of characters in the string. It MUST be aligned to a 2-byte boundary.

**Characters (variable):** A variable-length array of 16-bit Unicode values ordered from the beginning to the end of the string. It MUST be aligned to a 2-byte boundary. There is no special terminating character. The length of the array is the value of the **Number of Characters** field. It MUST terminate at a 2-byte boundary.

2.2.3.1.1.2 String List

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 1 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 2 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 3 | 0 | 1 |

Number of Strings

String Records (variable)

...
**Number of Strings (4 bytes):** A 32-bit unsigned integer representing the number of strings in the list. It MUST be located at the beginning of the file.

**String Records (variable):** A variable-length array of string records as specified in section 2.2.3.1.1. The number of string records in the array is the value of the Number of Strings field. It MUST terminate at a 2-byte boundary.

2.2.3.2 Versioned Index Identifier

This is a 32-bit unsigned integer associated with one full-text index component.

<table>
<thead>
<tr>
<th>0 1 2 3 4 5 6 7 8 9 1 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved0</td>
<td>Format Version</td>
</tr>
</tbody>
</table>

**Reserved0 (1 byte):** Must be set to "0x00".

**Format Version (1 byte):** An 8-bit unsigned integer value that MUST be "0x54", if the format version of the full-text index component, as specified in [MS-CIFO] section 2.17, is "0x54". In all other cases, the value MUST be "0x01".

**Reserved1 (1 byte):** Must be set to "0x00".

**Index ID (1 byte):** An 8-bit unsigned integer equal to the index identifier of the full-text index component, as specified in [MS-CIFO] section 2.17.

2.2.4 Result Sets

All results sets are specified in section 3.1.5.

2.2.5 Tables and Views

None.

2.2.6 XML Structures

None.

2.2.6.1 Namespaces

None.

2.2.6.2 Simple Types

None.

2.2.6.3 Complex Types

None.

2.2.6.4 Elements

None.
2.2.6.5 Attributes
None.

2.2.6.6 Groups
None.

2.2.6.7 Attribute Groups
None.
3 Protocol Details

There are three roles of this protocol, as follows:

- Back-end database server
- Sender
- Receiver

A sender has a one-to-one correspondence with a crawl component, and a receiver has a one-to-one correspondence with a query component (2).

All back-end database server, sender, and receiver state, as specified in this section, applies to one search application only. Only crawl components and query components (2) from that particular search application are represented as senders and receivers in the following sections.

The execution of this protocol for one search application is completely independent from its execution for another search application. Even if a crawl component or query component (2) from another search application is present on the same server, its behavior is specified by the same rules but with completely separate state.

Most of the messages sent and received in this protocol are stored procedure calls and the result sets they return. These stored procedure and result set messages are specified in section 3.1.5. There is one non-stored procedure related message, a file transfer from sender to receiver, that is specified in section 2.2.3.1.

3.1 Back-End Database Server Details

The back-end database server responds to stored procedure calls from the sender and the receiver. It returns result sets and return codes and never initiates communication with either the sender or the receiver.

3.1.1 Abstract Data Model

The following section specifies data and state that are sufficient to specify the behavior of the back-end database server. The only state necessary for execution of this protocol from the back-end database server is a list of ready query components, as specified in section 3.1.1.1, and a list of running tasks, as specified in section 3.1.1.2.

3.1.1.1 List of Ready Query Components

The list of all query components (2) in the query component set, as specified in [MS-SRCHTP] section 3.1.1.2, in which the State value, as specified in [MS-SRCHTP] section 3.1.1.2, is either "Ready" or "IndexSplitDone", as specified in [MS-SRCHTP] section 2.2.1.3, and are members of the query topology whose State value is "Active", as specified in [MS-SRCHTP] section 2.2.1.2.

3.1.1.2 List of Running Tasks

A list of zero or more running tasks. Each running task represents one propagation task that is currently being performed by all query components (2). A running task has the following properties.

- **taskType**: The task type, as specified in section 2.2.1.1, of the propagation task.
- **senderID**: The CrawlComponentNumber, as specified in [MS-SRCHTP] section 3.1.1.3, of the crawl component which created the propagation task.
catalogID: The catalog identifier, a specified in section 2.2.1.2, of the full-text index catalog to which the propagation task applies.

list of completions: A list of zero or more query components (2) that have finished the propagation task.

For component additions, a running task also has the following properties.

objectId: The versioned index identifier, as specified in section 2.2.3.2 of the full-text index component being added.

maxDocID: The maximum document identifier (1) of the full-text index component being added.

birthDate: The component birth date of the full-text index component being added.

time: The UTC time when the task was added to this list.

3.1.1.3 List of Propagation Errors

A list of zero or more propagation errors that have been reported in response to conditions encountered in the execution of this protocol. A propagation error has the following properties.

senderID: The CrawlComponentNumber, as specified in [MS-SRCHTP] section 3.1.1.3, of the crawl component that reported the propagation error, or NULL if the error was reported by a query component (2).

receiverID: The QueryComponentNumber, as specified in [MS-SRCHTP] section 3.1.1.2, of the query component (2) that reported the propagation error, or the query component (2) to which full-text index components could not be copied by a crawl component.

catalogID: The catalog identifier, as specified in section 2.2.1.2, of the full-text index catalog to which the propagation error applies.

type: The propagation error type, as specified in section 2.2.1.3, of the error.

message: Descriptive text about the error.

firstUtcTime: The UTC time when the error was first added to the list.

latestUtcTime: The latest UTC time when an error with the same propagation error type, as specified in section 2.2.1.3, was reported by the same sender or receiver.

3.1.2 Timers

None.

3.1.3 Initialization

Listening endpoints are set up on the back-end database server to handle inbound Tabular Data Stream (TDS) requests, as specified in [MS-TDS].

Authentication of the TDS connection to the back-end database server MUST occur before this protocol can be used.

The data structures, stored procedures, and actual data are persisted by the back-end database server within databases, so any operations to initialize the state of the database MUST occur before
the back-end database server can use this protocol. This protocol requires that the search administration data already exist within the back-end database server.

3.1.4 Higher-Layer Triggered Events

None.

3.1.5 Message Processing Events and Sequencing Rules

The back-end database server MUST NOT initiate any communication. It MUST only issue messages to other servers as result sets and return values, in direct response to incoming stored procedure calls.

There are no preconditions of state to receiving any of these calls; the back-end database server MUST be able to process them in any order, at any time after initialization.

As an aid to understanding, there is a naming convention for all of the propagation-related stored procedures. Procedures beginning with the prefix "proc_MSS_PropagationIndexer" are called from the sender. Procedures beginning with "proc_MSS_PropagationQueryComponent" are called from the receiver.

3.1.5.1 proc_MSS_PropagationIndexerCleanUpTablesForTask

The proc_MSS_PropagationIndexerCleanUpTablesForTask stored procedure is called to remove all records related to a finished propagation task. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_MSS_PropagationIndexerCleanUpTablesForTask(
    @SenderID           int,
    @CatalogID          int,
    @TaskType           int,
    @ObjectID           int
);
```

@SenderID: The sender identifier, as specified in section 3.2.1.2, of the sender that created the propagation task.

@CatalogID: The catalog identifier, as specified in section 2.2.1.2, representing the full-text index catalog to which the propagation task applies.

@TaskType: The task type, as specified in section 2.2.1.1, of the propagation task.

@ObjectID: If @TaskType is "ComponentAddition", as specified in section 2.2.1.1, @ObjectID MUST be either zero or the versioned index identifier, as specified in section 2.2.3.2, of the full-text index component that was propagated, as specified in section 3.2.5.9. If @TaskType is "StaticRankComputation", as specified in section 2.2.1.1, @ObjectID MUST be zero.

When the back-end database server receives this message, it does the following:

- If the State, as specified in [MS-SRCHTP] section 3.1.1.3, of the crawl component that created the propagation task is either "Disabled" or "DisableForRemove", as specified in [MS-SRCHTP] section 2.2.1.7, it MUST do nothing.
Otherwise, it MUST remove any propagation task from the list of running tasks where senderID equals @SenderID, catalogID equals @CatalogID, taskType equals @TaskType, and objectID equals @ObjectID. These parameters are specified in section 3.1.1.2.

The following table lists return code values.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>The task was added to the list of running tasks.</td>
</tr>
<tr>
<td>1</td>
<td>The task was not removed from the list of running tasks because the crawl component was disabled.</td>
</tr>
</tbody>
</table>

**Result Sets:** MUST NOT return any result sets.

### 3.1.5.2 proc_MSS_PropagationIndexerGetCompletedTasks

The proc_MSS_PropagationIndexerGetCompletedTasks stored procedure is called to retrieve every propagation task for a specified full-text index catalog that has been finished by all query components (2) in the list of ready query components, as specified in section 3.1.1.1. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_MSS_PropagationIndexerGetCompletedTasks(
    @SenderID           int,
    @CatalogID          int
);
```

@SenderID: The sender identifier, as specified in section 3.2.1.2, of the sender that created the propagation task.

@CatalogID: The catalog identifier, as specified in section 2.2.1.2, of the full-text index catalog for which the sender receives finished propagation tasks.

**Return Code Values:** An integer that MUST be zero.

**Result Sets:** MUST return the result set specified in section 3.1.5.2.1.

- If the State value, as specified in [MS-SRCHTP] section 3.1.1.3, of the crawl component that created the propagation task is either "Disabled" or "DisableForRemove", as specified in [MS-SRCHTP] section 2.2.1.7, the returned result set MUST contain zero results.
- Otherwise, the returned result set MUST contain exactly one result for each propagation task that was finished by all query components (2) in the list of ready query components, as specified in section 3.1.1.1, and MUST NOT contain other results.

### 3.1.5.2.1 Completed Tasks Result Set

The T-SQL syntax for the result set is as follows:

```sql
SenderID            int,
CatalogID           int,
TaskType            int,
ObjectId            int,
{MaxWorkID}         int,
{BirthDate}         int
```
SenderID: The senderID, as specified in section 3.1.2, of the propagation task.

CatalogID: The catalogID, as specified in section 3.1.2, of the propagation task.

TaskType: The taskType, as specified in section 3.1.2, of the propagation task.

ObjectId: If taskType is "ComponentAddition", as specified in section 2.2.1.1, ObjectId MUST be the objectID, as specified in section 3.1.2, of the running propagation task. If taskType is "StaticRankComputation", as specified in section 2.2.1.1, ObjectId MUST be zero.

{MaxWorkID}: An obsolete field that MUST be zero.

{BirthDate}: An obsolete field that MUST be zero.

3.1.5.3 proc_MSS_PropagationIndexerGetReadyQueryComponents

The proc_MSS_PropagationIndexerGetReadyQueryComponents stored procedure is called to retrieve information about all query components (2) in the list of ready query components, as specified in section 3.1.1.1. The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_PropagationIndexerGetReadyQueryComponents();
```

Return Code Values: An integer that MUST be zero.

Result Sets: MUST return exactly one ready query components result set, as specified in section 3.1.5.3.1. This result set MUST contain exactly one result for each query component (2) in the list of ready query components, as specified in section 3.1.2, and MUST NOT contain other results.

3.1.5.3.1 Ready Query Components Result Set

The T-SQL syntax for the result set is as follows:

```
ServerName              nvarchar(256),
QueryComponentNumber    int,
PartitionID             uniqueidentifier,
ShareName               nvarchar(260)
```

ServerName: The ServerName, as specified in [MS-SRCHTP] section 3.1.2, of the query component (2).

QueryComponentNumber: The QueryComponentNumber, as specified in [MS-SRCHTP] section 3.1.2, of the query component (2).

PartitionID: The PartitionID, as specified in [MS-SRCHTP] section 3.1.2, of the query component (2).

ShareName: The ShareName, as specified in [MS-SRCHTP] section 3.1.2, of the query component (2).

3.1.5.4 proc_MSS_PropagationIndexerGetTasks

The proc_MSS_PropagationIndexerGetTasks stored procedure is called to retrieve every propagation task that was created by the calling crawl component for a specified full-text index catalog. The T-SQL syntax for the stored procedure is as follows:
PROCEDURE proc_MSS_PropagationIndexerGetCompletedTasks(
    @SenderID    int,
    @CatalogID   int
);

@SenderID: The sender identifier, as specified in section 3.2.1.2, of the calling sender.

@CatalogID: The catalog identifier, as specified in section 2.2.1.2, of the full-text index catalog for which the caller receives propagation tasks.

Return Code Values: An integer that MUST be zero.

Result Sets: MUST return a propagation tasks result set, as specified in section 3.1.5.4.1. The returned result set MUST contain exactly one result for each propagation task in the list of running tasks, as specified in section 3.1.1.2, and MUST NOT contain other results.

3.1.5.4.1 Propagation Tasks Result Set

The T-SQL syntax for the result set is as follows:

SenderID    int,
CatalogID   int,
TaskType    int,
ObjectID    int,
MaxWorkID   int,
BirthDate   int

SenderID: The senderID, as specified in section 3.1.1.2, of the propagation task.

CatalogID: The catalogID, as specified in section 3.1.1.2, of the propagation task.

TaskType: The taskType, as specified in section 3.1.1.2, of the propagation task.

ObjectID: If taskType is "ComponentAddition", as specified in section 2.2.1.1, ObjectID MUST be the objectID, as specified in section 3.1.1.2 of the running propagation task. If taskType is "StaticRankComputation", as specified in section 2.2.1.1, ObjectID MUST be zero.

MaxWorkID: An obsolete field that MUST be zero.

BirthDate: An obsolete field that MUST be zero.

3.1.5.5 proc_MSS_PropagationIndexerInsertNewTask

The proc_MSS_PropagationIndexerInsertNewTask stored procedure is called to add a new propagation task to the list of running tasks, as specified in section 3.1.1.2. The T-SQL syntax for the stored procedure is as follows:

PROCEDURE proc_MSS_PropagationIndexerInsertNewTask(
    @SenderID    int,
    @CatalogID   int,
    @TaskType    int,
    @ObjectID    int,
    @MaxWorkID   int,
    @BirthDate   int
);
@SenderID: The sender identifier, as specified in section 3.2.1, of the calling sender.

@CatalogID: The catalog identifier, as specified in section 2.2.1, of the full-text index catalog to which the propagation task applies.

@TaskType: The task type, as specified in section 2.2.1, of the propagation task.

@ObjectID: If @TaskType is "ComponentAddition", as specified in section 2.2.1, @ObjectID MUST be either zero or the versioned index identifier, as specified in section 2.2.3, of the full-text index component that is being propagated, as specified in section 3.2.5.6. If @TaskType is "StaticRankComputation", as specified in section 2.2.1, @ObjectID MUST be zero.

@MaxWorkID: If @TaskType is "ComponentAddition", @MaxWorkID MUST be either zero or the maximum document identifier (1) in the full-text index component being propagated, as specified in section 3.2.5.6. If @TaskType is not "ComponentAddition", @MaxWorkID MUST be zero.

@BirthDate: If @TaskType is "ComponentAddition", @BirthDate MUST be either zero or the component birth date of the full-text index component being propagated, as specified in section 3.2.5.6. If @TaskType is not "ComponentAddition", @BirthDate MUST be zero.

When the back-end database server receives this message:

- If the value of State, as specified in [MS-SRCHTP] section 3.1.1.3, of the crawl component that created the propagation task is either "Disabled" or "DisableForRemove", as specified in [MS-SRCHTP] section 2.2.1.7, the back-end database server MUST return 2.

- Otherwise,
  - If the @CatalogID, @TaskType, and @ObjectID parameters match the catalogID, taskType, and objectID values of a propagation task in the list of running tasks, as specified in section 3.1.1.2, the back-end database server MUST return 1.
  - Otherwise, the back-end database server MUST add a new propagation task to the list of running tasks, where catalogID, as specified in section 3.1.1.2, equals @CatalogID, taskType, as specified in section 3.1.1.2, equals @TaskType, objectID, as specified in section 3.1.1.2, equals @ObjectID, and time, as specified in section 3.1.1.2, equals the current local time in datetime format.

Return Code Values:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>The propagation task was added.</td>
</tr>
<tr>
<td>1</td>
<td>No propagation task was added because a duplicate propagation task already existed.</td>
</tr>
<tr>
<td>2</td>
<td>No propagation task was added because the crawl component was disabled.</td>
</tr>
</tbody>
</table>

Result Set: MUST NOT return any result set.

3.1.5.6 proc_MSS_PropagationQueryComponentPickUpNewPropagationItems

The proc_MSS_PropagationQueryComponentPickUpNewPropagationItems stored procedure is called to get information about all the running tasks for a particular full-text index catalog that
have not yet been finished by a particular query component (2). The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_MSS_PropagationQueryComponentPickUpNewPropagationItems(
    @CatalogID          int,
    @ReceiverID         int
);
```

**@CatalogID:** The catalog identifier, as specified in section 2.2.1.2, representing the full-text index catalog to which the retrieved propagation tasks apply.

**@ReceiverID:** The receiver identifier, as specified in section 3.3.1.1, of the calling receiver.

When the back-end database server receives this message, it does the following:

- If no query component (2) with @ReceiverID is in the list of ready query components, as specified in section 3.1.1.1, it MUST return 1.
- Otherwise, it MUST return zero.

**Return Code Values:**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>A result set with the incomplete propagation tasks was returned.</td>
</tr>
<tr>
<td>1</td>
<td>No result set was returned because the query component was not in the list of ready query components.</td>
</tr>
</tbody>
</table>

**Result Sets:**

If no query component (2) with @ReceiverID is in the list of ready query components, the back-end database server MUST NOT return any result sets.

Otherwise, it MUST return exactly one propagation tasks result set, as specified in section 3.1.5.6.1. The result set MUST include exactly one result for each propagation task in the list of running tasks, as specified in section 3.1.1.2, in which the catalogID, as specified in section 3.1.1.2, is @CatalogID and whose list of completions, as specified in section 3.1.1.2, does not contain any query component with a QueryComponentNumber, as specified in [MS-SRCHTP] section 3.1.1.2, equal to @ReceiverID. The result set MUST NOT include any other results. The results MUST be ordered in ascending order, primarily by senderID values, as specified in section 3.1.5.6.1, and secondarily by birthDate values, as specified in section 3.1.5.6.1.

**3.1.5.6.1 Propagation Tasks Result Set**

The T-SQL syntax for the result set is as follows:

```sql
SenderID            int,
CatalogID           int,
TaskType            int,
ObjectID            int,
MaxWorkID           int,
BirthDate           int
```

**SenderID:** The senderID, as specified in section 3.1.1.2, of the propagation task.
CatalogID: The catalogID, as specified in section 3.1.1.2, of the propagation task. This MUST be the same value as the input parameter @CatalogID.

TaskType: The taskType, as specified in section 3.1.1.2, of the propagation task.

ObjectID: For all results where the value of TaskType is "ComponentAddition", as specified in section 2.2.1.1, ObjectID MUST be the objectID, as specified in section 3.1.1.2, of the propagation task. For all results where the value of TaskType is not "ComponentAddition", ObjectID MUST be zero.

MaxWorkID: For all results where the value of TaskType is "ComponentAddition", as specified in section 2.2.1.1, MaxWorkID MUST be the value of maxDocID, as specified in section 3.1.1.2, of the propagation task. For all results where the TaskType value is not "ComponentAddition", MaxWorkID MUST be zero.

BirthDate: For all results where the value of TaskType is "ComponentAddition", as specified in section 2.2.1.1, BirthDate MUST be the value of birthdate, as specified in section 3.1.1.2, of the propagation task. For all results where the value of TaskType is not "ComponentAddition", BirthDate MUST be zero.

3.1.5.7 proc_MSS_PropagationIndexerDeleteAllTasksFromSender

The proc_MSS_PropagationIndexerDeleteAllTasksFromSender stored procedure is called to delete all propagation tasks from the list of running tasks, as specified in section 3.1.1.2, that were created by the calling sender. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_MSS_PropagationIndexerDeleteAllTasksFromSender(
    @SenderID           int,
    @CatalogID          int
);
```

@SenderID: The sender identifier, as specified in section 3.2.1.2, of the calling sender.

@CatalogID: The catalog identifier, as specified in section 2.2.1.2, of the full-text index catalog for which the propagation tasks are removed from the list of running tasks, as specified in section 3.1.1.2.

When the back-end database server receives this message, it MUST delete all propagation tasks in the list of running tasks whose senderID value, as specified in section 3.1.1.2, is equal to @SenderID and catalogID value, as specified in section 3.1.1.2, is equal to @CatalogID.

Return Code Values: An integer that MUST be zero.

Result Sets: MUST NOT return any result set.

3.1.5.8 proc_MSS_PropagationQueryComponentReportTaskReady

The proc_MSS_PropagationQueryComponentReportTaskReady stored procedure is called to record that a query component (2) has finished processing a propagation task. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_MSS_PropagationQueryComponentReportTaskReady(
    @SenderID           int,
    @CatalogID          int,
    @ReceiverID         int,
);
```

[MS-CIPROP2] — v20120630
Index Propagation Version 2 Protocol Specification

Copyright © 2012 Microsoft Corporation.

Release: July 16, 2012
@TaskType int,
@ObjectID int,
);

@SenderID: The sender identifier, as specified in section 3.2.1.2, of the sender that created the propagation task.

@CatalogID: The catalog identifier, as specified in section 2.2.1.2, of the full-text index catalog to which the propagation task applies.

@ReceiverID: The receiver identifier, as specified in section 3.3.1.1, of the calling receiver.

@TaskType: Any value of task type, as specified in section 2.2.1.1.

@ObjectID: If @TaskType is "ComponentAddition", as specified in section 2.2.1.1, @ObjectID MUST be the versioned index identifier, as specified in section 2.2.3.2, of the full-text index component that is being propagated. If @TaskType is "StaticRankComputation", as specified in section 2.2.1.1, @ObjectID MUST be zero.

When the back-end database server receives this message, it does the following:

- If there is no query component (2) in the list of ready components, as specified in section 3.1.1.1, with QueryComponentNumber, as specified in [MS-SRCHTP] section 3.1.1.2, equal to @ReceiverID, the back-end database server MUST return 1.

- Otherwise,

  - If the query component (2) with QueryComponentNumber equal to @ReceiverID is already in the list of completions for the propagation task in the list of running tasks where catalogID equals @CatalogID, taskType equals @TaskType, and objectID equals @ObjectID, the back-end database server MUST return 1. The list of completions, list of running tasks, catalogID, taskType, and objectID are specified in section 3.1.1.

  - Otherwise, the back-end database server MUST add the query component (2) with QueryComponentNumber equal to @ReceiverID to the list of completions for the propagation task in the list of running tasks where catalogID equals @CatalogID, taskType equals @TaskType, and objectID equals @ObjectID. In addition, if @TaskType is "ComponentAddition", as specified in section 2.2.1.1, and @ObjectID is zero, the calling receiver MUST be removed from the list of completions for all tasks in the list of running tasks whose taskType is "ComponentAddition". The list of completions, list of running tasks, catalogID, taskType, and objectID are specified in section 3.1.1.

Return code values are listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful execution.</td>
</tr>
<tr>
<td>1</td>
<td>No change to the list of running tasks was made, because the receiver was not in the list of ready query components, or a completion for this task was already recorded for the receiver.</td>
</tr>
</tbody>
</table>

Result Set: MUST NOT return any result set.
3.1.5.9 proc_MSS_PropagationReportError

The proc_MSS_PropagationReportError stored procedure is called to add one propagation error to the list of propagation errors, as specified in section 3.1.1.3, if that error does not already exist, or to update the error's properties if it does. The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_PropagationReportError(
    @SenderID           int,
    @ReceiverID         int,
    @CatalogID          int,
    @Type               int,
    @Message            nvarchar(2048)
);
```

@SenderID: If @Type is "FileCopy", as specified in section 2.2.1.3, the sender identifier, as specified in section 3.2.1.2, of the calling sender. If @Type is "IndexCorruption" or @Type is "IndexAbsorption", the value is zero. If @Type is "ReceiverHang", it MUST be NULL.

@ReceiverID: If @Type is "FileCopy", as specified in section 2.2.1.3, the receiver identifier, as specified in section 3.3.1.1, of the receiver to which a full-text index component could not be copied. Otherwise, the receiver identifier of the receiver on which the error was encountered.

@CatalogID: The catalog identifier, as specified in section 2.2.1.2, of the full-text index catalog for which the propagation error is removed from the list of propagation errors, as specified in section 3.1.1.3.

@Type: The propagation error type, as specified in section 2.2.1.3, of the error that was encountered.

@Message: A string containing descriptive text about the error.

When the back-end database server receives this message, it does the following:

- If there is no error in the list of propagation errors, as specified in section 3.1.1.3, with the senderID value, as specified in section 3.1.1.3, equal to @SenderID, receiverID value, as specified in section 3.1.1.3, equal to @ReceiverID, catalogID value, as specified in section 3.1.1.3, equal to @CatalogID, and type, as specified in section 3.1.1.3, equal to @Type, it MUST change latestUtcTime, as specified in section 3.1.1.3, to the current time, and it MUST change message, as specified in section 3.1.1.3, to @Message.
- Otherwise, it MUST add a new propagation error to the list of propagation errors, as specified in section 3.1.1.3, where senderID is set to @SenderID, receiverID is set to @ReceiverID, catalogID is set to @CatalogID, message is set to @Message, and both firstUtcTime, as specified in section 3.1.1.3, and latestUtcTime are set to the current time.

Return Code Values: An integer that MUST be ignored.

Result Sets: MUST NOT return any result set.

3.1.5.10 proc_MSS_PropagationDeleteError

The proc_MSS_PropagationDeleteError stored procedure is called to delete one propagation error from the list of propagation errors, as specified in section 3.1.1.3. The T-SQL syntax for the stored procedure is as follows:
PROCEDURE proc_MSS_PropagationDeleteError(
  @SenderID           int,
  @ReceiverID         int,
  @CatalogID          int,
  @Type               int
);

@SenderID: If @Type is "FileCopy", as specified in section 2.2.1.3, the sender identifier, as specified in section 3.2.1.2, of the calling sender. If @Type is "IndexCorruption" or "IndexAbsorption", the value is zero. If @Type is "ReceiverHang", it MUST be NULL.

@ReceiverID: If @Type is "FileCopy", as specified in section 2.2.1.3, the receiver identifier, as specified in section 3.1.1.3, of the receiver to which a full-text index component could not be copied. Otherwise, the receiver identifier of the receiver on which the error was encountered.

@CatalogID: The catalog identifier, as specified in section 2.2.1.2, of the full-text index catalog for which the propagation error will be removed from the list of propagation errors, as specified in section 3.1.1.3.

@Type: The propagation error type, as specified in section 2.2.1.3, of the error to be removed.

When the back-end database server receives this message, it MUST delete the propagation error in the list of propagation errors, as specified in section 3.1.1.3, with the senderID value, as specified in section 3.1.1.3, equal to @SenderID, receiverID value, as specified in section 3.1.1.3, equal to @ReceiverID, and catalogID value, as specified in section 3.1.1.3, equal to @CatalogID, if such a propagation error exists.

Return Code Values: An integer that MUST be ignored.

Result Sets: MUST NOT return any result set.

3.1.5.11 proc_MSS_PropagationDeleteErrors

The proc_MSS_PropagationDeleteErrors stored procedure is called to delete all propagation errors from the list of propagation errors, as specified in section 3.1.1.3 that were encountered by a particular receiver. The T-SQL syntax for the stored procedure is as follows:

PROCEDURE proc_MSS_PropagationDeleteErrors(
  @ReceiverID         int
);

@ReceiverID: The receiver identifier, as specified in section 3.3.1.1, of the receiver for which all propagation errors are to be removed.

When the back-end database server receives this message, it MUST delete all propagation errors in the list of propagation errors, as specified in section 3.1.1.3, with the receiverID value, as specified in section 3.1.1.3, equal to @ReceiverID.

Return Code Values: An integer that MUST be ignored.

Result Sets: MUST NOT return any result set.
### 3.1.5.12 proc_MSS_PropagationGetErrors

The `proc_MSS_PropagationGetErrors` stored procedure is called to retrieve information about all propagation errors in the list of propagation errors, as specified in section 3.1.1.3. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_MSS_PropagationGetErrors();
```

**Return Code Values:** An `integer` that MUST be ignored.

**Result Sets:** MUST return exactly one propagation errors result set, as specified in section 3.1.5.12.1. This result set MUST contain exactly one result for each propagation error in the list of propagation errors, as specified in section 3.1.1.3, and MUST NOT contain other results.

#### 3.1.5.12.1 Propagation Errors Result Set

The T-SQL syntax for the result set is as follows:

```sql
SenderID    int,
ReceiverID  int,
CatalogID   int,
Type        int,
Message     nvarchar(2048),
FirstUtcTime datetime,
LatestUtcTime datetime,
RowID       int
```

- **SenderID:** The `senderID`, as specified in section 3.1.1.3, of the propagation error.
- **ReceiverID:** The `receiverID`, as specified in section 3.1.1.3, of the propagation error.
- **CatalogID:** The `catalogID`, as specified in section 3.1.1.2, of the propagation error.
- **Type:** The `type`, as specified in section 3.1.1.3, of the propagation error.
- **Message:** The `message`, as specified in section 3.1.1.3, of the propagation error.
- **FirstUtcTime:** The `firstUtcTime`, as specified in section 3.1.1.3, of the propagation error.
- **LatestUtcTime:** The `latestUtcTime`, as specified in section 3.1.1.3, of the propagation error.
- **RowID:** Any `integer`. This value MUST be ignored.

### 3.1.5.13 proc_MSS_PropagationGetTasks

The `proc_MSS_PropagationGetTasks` stored procedure is called to retrieve information about all propagation tasks in the list of running tasks, as specified in section 3.1.1.2. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_MSS_PropagationGetTasks();
```

**Return Code Values:** An `integer` that MUST be ignored.
**Result Sets**: MUST return exactly one propagation all tasks result set, as specified in section 3.1.5.13.1. This result set MUST contain exactly one result for each propagation task in the list of running tasks, as specified in section 3.1.1.2, and MUST NOT contain other results.

### 3.1.5.13.1 Propagation All Tasks Result Set

The T-SQL syntax for the result set is as follows:

```
SenderID            int,
CatalogID           int,
TaskType            int,
ObjectID            int,
MaxWorkID           int,
BirthDate           int,
Time                datetime
```

- **SenderID**: The `senderID`, as specified in section 3.1.1.2, of the propagation task.
- **CatalogID**: The `catalogID`, as specified in section 3.1.1.2, of the propagation task.
- **TaskType**: The `taskType`, as specified in section 3.1.1.2, of the propagation task.
- **ObjectID**: The `objectID`, as specified in section 3.1.1.2, of the propagation task.
- **MaxWorkID**: The `maxDocID`, as specified in section 3.1.1.2, of the propagation task.
- **BirthDate**: The `birthdate`, as specified in section 3.1.1.2, of the propagation task.
- **Time**: The `time`, as specified in section 3.1.1.2, of the propagation task.

### 3.1.5.14 `proc_MSS_PropagationGetTaskCompletions`

The `proc_MSS_PropagationGetTaskCompletions` stored procedure is called to retrieve information about all propagation tasks in the list of running tasks, as specified in section 3.1.1.2, that have been finished by one or more receivers. The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_PropagationGetTaskCompletions();
```

- **Return Code Values**: An integer that MUST be ignored.
- **Result Sets**: MUST return exactly one propagation task completions result set, as specified in section 3.1.5.14.1. This result set MUST contain exactly one result for each receiver in the list of completions, as specified in section 3.1.1.2, for each propagation task in the list of running tasks, as specified in section 3.1.1.2, and MUST NOT contain other results.

#### 3.1.5.14.1 Propagation Task Completions Result Set

The T-SQL syntax for the result set is as follows:

```
ReceiverID          int,
SenderID            int,
CatalogID           int,
TaskType            int,
ObjectID            int,
```
MaxWorkID int,
BirthDate int

**ReceiverID:** The receiverID, as specified in section 3.1.1.2, that has finished the propagation task.

**SenderID:** The senderID, as specified in section 3.1.1.2, of the propagation task.

**CatalogID:** The catalogID, as specified in section 3.1.1.2, of the propagation task.

**TaskType:** The taskType, as specified in section 3.1.1.2, of the propagation task.

**ObjectId:** The objectId, as specified in section 3.1.1.2, of the propagation task.

**MaxWorkID:** The maxDocID, as specified in section 3.1.1.2, of the propagation task.

**BirthDate:** The birthdate, as specified in section 3.1.1.2, of the propagation task.

### 3.1.6 Timer Events

None.

### 3.1.7 Other Local Events

None.

### 3.2 Sender Details

The sender is implemented by a crawl component. It initiates all propagation sequences.

#### 3.2.1 Abstract Data Model

The following section specifies data and state that are sufficient to specify the behavior of the sender. The data provided explains how the protocol behaves. Implementations do not need to adhere to this model as long as their server-to-server communication is consistent with what is specified in this document.

#### 3.2.1.1 Search Application Name

The name of the search service application that the crawl component belongs to.

#### 3.2.1.2 Sender Identification

An integer that uniquely identifies the sender. This MUST be equal to the CrawlComponentNumber of the sender, as specified in [MS-SRCHTP] section 3.1.1.3.

#### 3.2.1.3 List of Ready Query Components

A list of zero or more query components (2) to receive full-text index component messages, as specified in section 2.2.3.1. Each ready query component has the following properties.

**servername:** The ServerName of the query component (2), as specified in [MS-SRCHTP] section 3.1.1.2.
shareName: The ShareName of the query component (2), as specified in [MS-SRCHTP] section 3.1.1.2.

3.2.1.4 List of Running Tasks

A list of zero or more propagation tasks that have been issued by this sender. Each running task has the following properties.

- **taskType**: The task type, as specified in section 2.2.1.1, of the propagation task.

- **catalogID**: The catalog identifier, as specified in section 2.2.1.2, of the full-text index catalog to which the propagation task applies.

- **objectID**: If **taskType** is "ComponentAddition", the versioned index identifier, as specified in section 2.2.3.2, of the full-text index component being added on the query components (2). For all other values of **taskType**, the value MUST be zero.

- **lastPropagationTime**: The time of the most recent sending of the full-text index component message, as specified in section 3.2.5.3.

3.2.1.5 List of Completed Tasks

A list of zero or more completed tasks. Each completed task has the following properties.

- **catalogID**: The catalog identifier, as specified in section 2.2.1.2, of the full-text index catalog to which the propagation task applies.

- **taskType**: The task type of the propagation task, as specified in section 2.2.1.1.

- **objectID**: If **taskType** is "ComponentAddition", the versioned index identifier, as specified in section 2.2.3.2, of the full-text index component being added on the query components (2). For all other values of **taskType**, the value MUST not be used.

3.2.1.6 Error Possibly Exists

A Boolean value that is "false" if, and only if, this sender has deleted all propagation errors for this sender from the list propagation errors, as specified in section 3.1.1.3.

3.2.2 Timers

None.

3.2.3 Initialization

The "error possibly exists" value, as specified in section 3.2.1.6, MUST be set to "true".

3.2.4 Higher-Layer Triggered Events

None.
3.2.5  Message Processing Events and Sequencing Rules

3.2.5.1  Sending a proc_MSS_PropagationIndexerGetReadyQueryComponents Message

The sender SHOULD call the proc_MSS_PropagationIndexerGetReadyQueryComponents stored procedure, as specified in section 3.1.5.3, on a periodic basis. If it does not do this, the sender MUST use another method of accurately updating its list of ready query components, as specified in section 3.2.1.3, to match the back-end database server’s list of ready query components, as specified in section 3.1.1.3. No special action is required if there is a change in this list.

3.2.5.2  Receiving a Ready Query Components Result Set

This result set, as specified in section 3.1.5.3.1, is received automatically after calling the proc_MSS_PropagationIndexerGetReadyQueryComponents stored procedure, as specified in section 3.1.5.3. The sender MUST replace its current list of ready query components, as specified in section 3.2.1.3, with exactly one ready query component for each received result, with the following settings:

- serverName, as specified in section 3.2.1.3, is set to the value of ServerName, as specified in section 3.1.5.3.1, of the result.
- shareName, as specified in section 3.2.1.3, is set to the value of ShareName, as specified in section 3.1.5.3.1, of the result.

The sender MUST NOT add any other ready query components into its list of ready query components, as specified in section 3.2.1.3).

3.2.5.3  Sending a Full-Text Index Component Message

This is the first message of a propagation sequence for a component addition action.

This message MUST be sent either when a full-text index component is generated for a full-text index catalog on the sender, or when any propagation task in the list of running tasks, as specified in section 3.2.1.4, in which the value of lastPropagationTime is more than 5 minutes prior to the current time.

To send this message, the sender MUST perform the following actions:

1. The sender generates a full-text index component, as specified in [MS-CIFO] section 2.17, except that each file name contains an additional prefix, which must be a 0-prefixed, 4-digit hexadecimal representation of the sender identifier, as specified in section 3.2.1.2, plus a period. See section 4.1.2 for an example.

2. The sender generates a propagation list file, as specified in section 2.2.3.1.1, containing the file names of each of the files contained in the propagated full-text index component message other than the propagation list file itself. See section 4.1.2 for an example.

3. Using the file copying protocol specified in [MS-SMB], the sender copies the duplicated full-text index component files and the propagation list file, as specified in section 2.2.3.1.1, to a path relative to serverName, as specified in section 3.2.1.3, and shareName, as specified in section 3.2.1.3, of each ready query component in the list of ready query components, as specified in section 3.2.1.3. The destination path MUST be \<server>\<share>\<application>-query-<receiverID>\Projects\<catalog>\Indexer\CiFiles\<file>, where the following are true:
• <server> is the serverName, as specified in section 3.2.1.3.
• <share> is the shareName, as specified in section 3.2.1.3.
• <application> is the search application name, as specified in section 3.2.1.1.
• <receiverID> is the QueryComponentNumber, as specified in [MS-SRCHTP] section 3.1.1.2.
• <catalog> is one of the following:
  • "Portal_Content" if the full-text index catalog is the main catalog, as specified in [MS-CIFO] section 2.18.1.
  • "AnchorProject" if the full-text index catalog is the anchor text catalog, as specified in [MS-CIFO] section 2.18.2.
• <file> is the file name specified in steps 1 or 2, as appropriate.

4. If this message is sent because of the generation of a new full-text index component by the sender, and no errors occurred when performing the file copies, it MUST call the proc_MSS_PropagationIndexerInsertNewTask stored procedure, as specified in section 3.1.5.5. Also, if the "error possibly exists" value, as specified in section 3.2.1.6, is true, the sender MUST call the proc_MSS_PropagationDeleteError stored procedure, as specified in section 3.2.5.5.

5. If this message is sent because of the generation of a new full-text index component by the sender, and any error occurred when performing the file copies, this message MUST be resent beginning at step 1. In addition, if more than five minutes have elapsed since the error was first encountered, the sender MUST call the proc_MSS_PropagationReportError stored procedure, as specified in section 3.2.5.4.

6. If this message is sent because the value of lastPropagationTime, as specified in section 3.2.1.4, of a propagation task in the list of running tasks, as specified in section 3.2.1.4, is at least 5 minutes prior to the current time, and no errors occurred when performing the file copies, the sender MUST update its lastPropagationTime value to the current time.

3.2.5.4 Sending a proc_MSS_PropagationReportError Message

The sender calls the proc_MSS_PropagationReportError stored procedure, as specified in section 3.1.5.10, after it has encountered an error while sending a full-text index component message, as specified in section 3.2.5.3, with the following parameter values:

• @SenderId MUST be the sender identifier, as specified in section 3.2.1.2, of this sender.
• @ReceiverID MUST be the receiver identifier, as specified in section 3.2.5.3, of the receiver to which the full-text index catalog component could not be copied.
• @CatalogID MUST be the identifier of the full-text index catalog whose full-text index catalog component could not be copied.
• @Type MUST be "FileCopy", as specified in section 2.2.1.3.
• @Message SHOULD be a string that gives additional description of the error encountered, but any string is allowed by the protocol.

After successfully sending this message, the "error possibly exists" value, as specified in section 3.2.1.6, SHOULD be set to "true".
3.2.5.5 Sending a proc_MSS_PropagationDeleteError Message

The sender calls the **proc_MSS_PropagationDeleteError** stored procedure, as specified in section 3.1.5.10, after it has successfully sent a full-text index component message, as specified in section 3.2.5.3, with the following parameter values:

- @SenderID MUST be the sender identifier, as specified in section 3.2.1.2, of this sender.
- @ReceiverID MUST be the receiver identifier, as specified in section 3.2.5.3, of the receiver to which the full-text index catalog component was successfully copied.
- @CatalogID MUST be the identifier of the full-text index catalog whose full-text index catalog component was successfully copied.
- @Type MUST be "FileCopy", as specified in section 2.2.1.3.

After successfully sending this message, the "error possibly exists" value, as specified in section 3.2.1.6, SHOULD be set to "false".

3.2.5.6 Sending a proc_MSS_PropagationIndexerInsertNewTask Message

This is the first message of the propagation sequence for cleaning, which is specified in the following sections, and static rank computation, and the second message of the propagation sequence for component addition actions.

For any of the following events, the receiver calls the **proc_MSS_PropagationIndexerInsertNewTask** stored procedure, as specified in section 3.1.5.5:

- Component addition. The sender calls the **proc_MSS_PropagationIndexerInsertNewTask** stored procedure with the following parameters:
  - @SenderID MUST be the sender identifier, as specified in section 3.2.1.2, of this sender.
  - @CatalogID MUST be the catalog identifier, as specified in section 2.2.1.2, for the full-text index catalog.
  - @TaskType MUST be "ComponentAddition", as specified in section 2.2.1.1.
  - @ObjectID MUST be the versioned index identifier, as specified in section 2.2.3.2, of the full-text index component.
  - @MaxWorkID MUST be the maximum document identifier (1) in the full-text index component.
  - @BirthDate MUST be the component birth date of the document set of the full-text index component.

- Cleaning. It is often desirable to ensure that all query components (2) have finished all tasks before inserting another one. For this, the sender SHOULD call the **proc_MSS_PropagationIndexerInsertNewTask** stored procedure with the following parameters:
  - @SenderID MUST be the sender identifier, as specified in section 3.2.1.2, of this sender.
  - @CatalogID MUST be the catalog identifier, as specified in section 2.2.1.2, for the full-text index catalog.
@TaskType MUST be "ComponentAddition", as specified in section 2.2.1.

@ObjectID MUST be zero.

@MaxWorkID MUST be zero.

@BirthDate MUST be zero.

No change in the behavior of the sender is necessary if it does not send this message.

- Static rank computation. The sender calls the proc_MSS_PropagationIndexerInsertNewTask stored procedure with the following parameters:
  - @SenderID MUST be the sender identifier, as specified in section 3.2.1.2.
  - @CatalogID MUST be the catalog identifier, as specified in section 2.2.1.2.
  - @TaskType MUST be "StaticRankComputation", as specified in section 2.2.1.2.
  - @ObjectID MUST be zero.
  - @MaxWorkID MUST be zero.
  - @BirthDate MUST be zero.

If the back-end database server returns the value zero or 1, the message was sent successfully. Otherwise the message was not sent successfully.

After successfully sending this message and receiving a return value of zero, a propagation task MUST be added to the list of running tasks, as specified in section 2.2.1.2, with taskType set to the @TaskType value of the message sent, catalogID set to the @CatalogID value of the message sent, objectID set to the @ObjectID value of the message sent, and lastPropagationTime set to the current time.

If the proc_MSS_PropagationIndexerInsertNewTask stored procedure could not be called successfully, the call MUST be retried using the same parameters. The retry interval SHOULD be at least 3 seconds. There MUST NOT be a limit on the number of retries.

### 3.2.5.7 Sending a proc_MSS_PropagationIndexerGetCompletedTasks Message

This stored procedure is called to retrieve information about any propagation tasks in the list of running tasks, as specified in section 3.1.1.2, that have been finished by all query components (2), so that they can be removed from the back-end database server's list of running tasks.

The sender MUST call the proc_MSS_PropagationIndexerGetCompletedTasks stored procedure, as specified in section 3.1.5.2, periodically, for both the main catalog, as specified in [MS-CIFO] section 2.18.1, and the anchor text catalog, as specified in [MS-CIFO] section 2.18.2. The time interval between calls SHOULD be between 3 and 30 seconds, but using another interval does not prevent the successful execution of propagation tasks.

- @SenderID, as specified in section 3.1.5.2, MUST be the sender identifier, as specified in section 3.2.1.2, of this sender.

- @CatalogID, as specified in section 3.1.5.2, MUST be the identifier of the full-text index catalog.

If the proc_MSS_PropagationIndexerGetCompletedTasks stored procedure could not be called successfully, the call MUST be retried using the same parameters. The retry interval SHOULD be at least 3 seconds. There MUST NOT be a limit on the number of retries.
3.2.5.8 Receiving a Completed Tasks Result Set Message

A completed tasks result set, as specified in section 3.1.5.2.1, is received automatically following any call to the `proc_MSS_PropagationIndexerGetCompletedTasks` stored procedure, as specified in section 3.1.5.2.2. The full-text index catalog to which the result set applies is evident in the value of `CatalogID`, as specified in section 3.1.5.4.1, of each result in the result set. For this full-text index catalog, the sender MUST replace its list of completed tasks, as specified in section 3.2.1.5, with a new list containing one completed task for each result in this result set, with the following settings:

- `catalogID`, as specified in section 3.2.1.5, is set to the value of `CatalogID`, as specified in section 3.1.5.2.1, of the result.

- `taskType`, as specified in section 3.2.1.5, is set to the value of `TaskType`, as specified in section 3.1.5.2.1, of the result.

- `objectID`, as specified in section 3.2.1.5, is set to the value of `ObjectID`, as specified in section 3.1.5.2.1, of the result.

The sender MUST also remove the propagation task with `catalogID` equal to the value of `CatalogID`, as specified in section 3.1.5.2.1, of the result, and `objectID` equal to the value of `ObjectID`, as specified in section 3.1.5.2.1, of the result, from the list of running tasks, as specified in section 3.2.1.4.

3.2.5.9 Sending a `proc_MSS_PropagationIndexerCleanUpTablesForTask` Message

This is the final message sent in the propagation sequence of any propagation task.

Whenever there is at least one propagation task in the list of completed tasks, as specified in section 3.2.1.5, the `proc_MSS_PropagationIndexerCleanUpTablesForTask` stored procedure, as specified in section 3.1.5.1, MUST be called once for each completed task, using the following parameters:

- `@CatalogID`, as specified in section 3.1.5.1, is set to the `catalogID` value, as specified in section 3.2.1.5, of the completed task.

- `@TaskType`, as specified in section 3.1.5.1, is set to the `taskType` value, as specified in section 3.2.1.5, of the completed task.

- `@ObjectID`, as specified in section 3.1.5.1, is set to the `objectID` value, as specified in section 3.2.1.5, of the completed task.

After successfully sending this message, the sender MUST also remove the propagation task with `catalogID` equal to the `CatalogID` value, as specified in section 3.1.5.2.1, of the result, and `objectID` equal to the `ObjectID` value, as specified in section 3.1.5.2.1, of the result, from the list of completed tasks, as specified in section 3.2.1.5.

If the `proc_MSS_PropagationIndexerCleanUpTablesForTask` stored procedure could not be called successfully, the call MUST be retried using the same parameters. The retry interval SHOULD be at least 3 seconds. There MUST NOT be a limit on the number of retries.

3.2.5.10 Sending the `proc_MSS_PropagationIndexerGetTasks` Message

The sender calls the `proc_MSS_PropagationIndexerGetTasks` stored procedure, as specified in section 3.1.5.4, at any time, with no precondition. Processes on the sender use this to get the back-end database server's list of all running tasks, as specified in section 3.1.1.2. The `@SenderID`
parameter, as specified in section 3.1.5.4, MUST be the sender's sender identifier, as specified in section 3.2.1.2. The @CatalogID parameter, as specified in section 3.1.5.4, MUST be the catalog identifier, as specified in section 2.2.1.2, of a full-text index catalog.

If the proc_MSS_PropagationIndexerGetTasks stored procedure could not be called successfully, the call MUST be retried using the same parameters. The retry interval SHOULD be at least 3 seconds. There MUST NOT be a limit on the number of retries.

3.2.5.11 Receiving a Propagation Tasks Result Set

This result set is received automatically after calling the proc_MSS_PropagationIndexerGetTasks stored procedure, as specified in section 3.2.5.10. Receiving this message MUST NOT affect the state of the sender that is specified in section 3.2.1.

3.2.5.12 Sending the proc_MSS_PropagationIndexerDeleteAllTasksFromSender Message

The sender calls the proc_MSS_PropagationIndexerDeleteAllTasksFromSender stored procedure, as specified in section 3.1.5.7, at any time, with no precondition, using the following parameters:

- @SenderID, as specified in section 3.1.5.7, is set to the senderID value, as specified in section 3.2.1.2, of the sender.
- @CatalogID, as specified in section 3.1.5.7, is set to the catalogID value, as specified in section 2.2.1.2, of the full-text index catalog for which the propagation tasks are to be removed from the list of running tasks, as specified in section 3.1.1.2.

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

3.3 Receiver Details

The receiver is implemented by a query component (2). A receiver uses the protocol to apply changes to its full-text index catalogs and to perform static rank computation on its full-text index catalogs.

3.3.1 Abstract Data Model

The following section specifies data and state that are sufficient to specify the behavior of the receiver. Implementations do not need to adhere to this model as long as their server-to-server communication is consistent with that which is specified in this document.

3.3.1.1 Receiver Identifier

An integer that uniquely identifies the receiver. This MUST be equal to the QueryComponentNumber of the query component (2), as specified in [MS-SRCHTP] section 3.1.1.2.
3.3.1.2 List of Incomplete Tasks

A list of zero or more incomplete tasks. An incomplete task has the following properties.

senderID: The sender identifier, as specified in section 3.2.1.2, of the sender that created the propagation task.

catalogID: The catalog identifier, as specified in section 2.2.1.2, of the full-text index catalog to which the propagation task applies.

taskType: The task type of the propagation task, as specified in section 2.2.1.1.

objectID: If taskType is "ComponentAddition", as specified in section 2.2.1.1, the versioned index identifier, as specified in section 2.2.3.2, of the full-text index component being added on the query components (2). For all other values of taskType, the value is not used.

maxWorkID: If taskType is "ComponentAddition", as specified in section 2.2.1.1, the maximum document identifier (1) in the full-text index component. For all other values of taskType, the value is not used.

birthDate: If taskType is "ComponentAddition", as specified in section 2.2.1.1, the component birth date of the document set of the full-text index component. For all other values of taskType, the value is not used.

3.3.1.3 Error Possibly Exists

A Boolean value that is "false" if, and only if, this receiver has deleted all propagation errors for this receiver from the list propagation errors, as specified in section 3.1.1.3.

3.3.2 Timers

None.

3.3.3 Initialization

The "error possibly exists" value, as specified in section 3.3.1.3, MUST be set to "true".

3.3.4 Higher-Layer Triggered Events

None.

3.3.5 Message Processing Events and Sequencing Rules

3.3.5.1 Sending a proc_MSS_PropagationQueryComponentPickUpNewPropagationItems Message

All activity on a query component (2) for a propagation sequence begins with this call.

The receiver MUST call the proc_MSS_PropagationQueryComponentPickUpNewPropagationItems stored procedure, as specified in section 3.1.5.6, periodically, for both the main catalog, as specified in [MS-CIFO] section 2.18.1, and the anchor text catalog, as specified in [MS-CIFO] section 2.18.2. The time interval between calls SHOULD be between 3 and 30 seconds, but using another interval does not prevent the successful execution of propagation tasks. The procedure MUST be called with the following parameters:
- @ReceiverID, as specified in section 3.1.5.6, MUST be the receiver identifier, as specified in section 3.3.1, of this receiver.

- @CatalogID, as specified in section 3.1.5.6, MUST be the identifier of the full-text index catalog.

If the proc_MSS_PropagationQueryComponentPickUpNewPropagationItems stored procedure could not be called successfully, the call MUST be retried using the same parameters. The retry interval SHOULD be at least 3 seconds. There MUST NOT be a limit on the number of retries.

### 3.3.5.2 Receiving a Propagation Tasks Result Set

A propagation tasks result set is received automatically after sending a proc_MSS_PropagationQueryServerPickUpNewPropagationItems message, as specified in section 3.1.5.6. The full-text index catalog to which the result set applies is specified in the value of CatalogID, as specified in section 3.1.5.4.1, of each result in the result set. For this full-text index catalog, the query component (2) MUST replace its list of incomplete tasks, as specified in section 3.3.1.2, with a new list that contains one incomplete task for each result in this result set, where the following are true:

- senderID, as specified in section 3.3.1.2, is set to the value of SenderID, as specified in section 3.1.5.6, of the result.

- catalogID, as specified in section 3.3.1.2, is set to the value of CatalogID, as specified in section 3.1.5.6, of the result.

- taskType, as specified in section 3.3.1.2, is set to the value of TaskType, as specified in section 3.1.5.6, of the result.

- objectID, as specified in section 3.3.1.2, is set to the value of ObjectID, as specified in section 3.1.5.6, of the result.

- maxWorkID, as specified in section 3.3.1.2, is set to the value of MaxWorkID, as specified in section 3.1.5.6, of the result.

- birthdate, as specified in section 3.3.1.2, is the value of BirthDate, as specified in section 3.1.5.6, of the result.

### 3.3.5.3 Sending a proc_MSS_PropagationReportError Message

If the receiver encounters an error while processing the propagation task, it MUST call the proc_MSS_PropagationReportError stored procedure, as specified in section 3.1.5.10, with the following parameter values:

- @SenderID MUST be zero.

- @ReceiverID MUST be the receiver identifier, as specified in section 3.3.1.1.

- @CatalogID MUST be the identifier of the full-text index catalog whose full-text index catalog component could not be absorbed.

- @Type MUST be "IndexAbsorption", as specified in section 2.2.1.3.

- @Message SHOULD be a string that gives additional description of the error encountered, but any string is allowed by the protocol.

After successfully sending this message, the "error possibly exists" value, as specified in section 3.3.1.3, should be set to "true".
The receiver also calls the **proc_MSS_PropagationReportError** stored procedure, as specified in section **3.1.5.10**, after it has encountered any data that is not valid in the full-text index catalog, with the following parameter values:

- @SenderID MUST be zero.
- @ReceiverID MUST be the receiver identifier, as specified in section **3.3.1.1**.
- @CatalogID MUST be the identifier of the full-text index catalog that has been found to contain data that is not valid.
- @Type MUST be "IndexCorruption", as specified in section **2.2.1.3**.
- @Message MUST be text that gives additional description of the error encountered.

### 3.3.5.4 Sending a proc_MSS_PropagationDeleteError Message

If the receiver has successfully processed a propagation task, and the "error possibly exists" value, as specified in section **3.3.1.3**, is "true", the receiver MUST call the **proc_MSS_PropagationDeleteError** stored procedure, as specified in section **3.1.5.10**, with the following parameter values:

- @SenderID MUST be zero.
- @ReceiverID MUST be the receiver identifier, as specified in section **3.3.1.1**.
- @CatalogID MUST be the identifier of the full-text index catalog with the full-text index catalog component that could not be absorbed.
- @Type MUST be "IndexAbsorption", as specified in section **2.2.1.3**.

After successfully sending this message, the "error possibly exists" value, as specified in section **3.3.1.3**, should be set to "false".

### 3.3.5.5 Sending a proc_MSS_PropagationQueryComponentReportTaskReady Message

This message is the last message sent by a receiver in the propagation sequence.

The **proc_MSS_PropagationQueryComponentReportTaskReady** stored procedure, as specified in section **3.1.5.8**, MUST be called once for each incomplete task in the receiver's list of incomplete tasks, as specified in section **3.3.1.2**, with the following parameter values:

- @SenderID, as specified in section **3.1.5.8**, is set to the value of senderID, as specified in section **3.3.1.1**.
- @ReceiverID, as specified in section **3.1.5.8**, is set to the receiver identifier, as specified in section **3.3.1.1**.
- @CatalogID, as specified in section **3.1.5.8**, is set to the value of catalogID, as specified in section **3.3.1.2**.
- @TaskType, as specified in section **3.1.5.8**, is set to the value of taskType, as specified in section **3.3.1.2**.
- @ObjectID, as specified in section **3.1.5.8**, is set to the value of objectID, as specified in section **3.3.1.2**.
If the proc_MSS_PropagationQueryComponentReportTaskReady stored procedure could not be called successfully, the call MUST be retried using the same parameters. The retry interval SHOULD be at least 3 seconds. There MUST NOT be a limit on the number of retries.

3.3.6 Timer Events

None.

3.3.7 Other Local Events

None.

3.4 Admin Server Details

The admin server MUST be implemented on the server where the administration component exists. The admin server periodically interprets the current list of propagation errors and takes query components (2) offline when appropriate.

3.4.1 Abstract Data Model

The following section specifies data and state that are sufficient to specify the behavior of the admin server. Implementations do not need to adhere to this model as long as their server-to-server communication is consistent with that which is specified in this document.

3.4.1.1 List of Running Tasks

A list of zero or more running tasks. Each running task represents one propagation task that is currently being performed by all query components (2). A running task has the following properties.

taskId: The task type, as specified in section 2.2.1.1, of the propagation task.

senderID: The CrawlComponentNumber, as specified in [MS-SRCHTP] section 3.1.1.3, of the crawl component that created the propagation task.

catalogID: The catalog identifier, as specified in section 2.2.1.2, of the full-text index catalog to which the propagation task applies.

objectID: The versioned index identifier, as specified in section 2.2.3.2, of the full-text index component being added.

maxDocID: The maximum document identifier (1) of the full-text index component being added.

birthDate: The component birth date of the full-text index component being added.

time: The UTC time when the task was added to this list.

3.4.1.2 List of Task Completions

A list of zero or more records of the completion of a propagation task by a particular receiver. A task completion has the following properties.

receiverID: The QueryComponentNumber, as specified in [MS-SRCHTP] section 3.1.1.2, of the query component (2) that finished the propagation task.

taskType: The task type, as specified in section 2.2.1.1, of the propagation task.
**senderID**: The *CrawlComponentNumber*, as specified in [MS-SRCHTP] section 3.1.1.3, of the crawl component which created the propagation task.

**catalogID**: The catalog identifier, as specified in section 2.2.2, of the full-text index catalog to which the propagation task applies.

**objectID**: The versioned index identifier, as specified in section 2.2.3, of the full-text index component being added.

**maxDocID**: The maximum document identifier (1) of the full-text index component being added.

**birthDate**: The component birth date of the full-text index component being added.

### 3.4.1.3 List of Propagation Errors

A list of zero or more propagation errors that have been reported in response to conditions encountered in the execution of this protocol. A propagation error has the following properties.

**senderID**: The *CrawlComponentNumber*, as specified in [MS-SRCHTP] section 3.1.1.3, of the crawl component that reported the propagation error, or NULL if the error was reported by a query component (2).

**receiverID**: The *QueryComponentNumber*, as specified in [MS-SRCHTP] section 3.1.1.2, of the query component that reported the propagation error, or the query component (2) to which full-text index components could not be copied by a crawl component.

**catalogID**: The catalog identifier, as specified in section 2.2.2.2, of the full-text index catalog to which the propagation error applies.

**type**: The propagation error type, as specified in section 2.2.1.3, of the error.

**firstUtcTime**: The UTC time when the error was first added to the list.

**latestUtcTime**: The latest UTC time when an error with the same propagation error type, as specified in section 2.2.1.3, was reported by the same sender or receiver.

### 3.4.1.4 List of Receivers

A list of receivers. A receiver has the following properties.

**receiverID**: The *QueryComponentNumber*, as specified in [MS-SRCHTP] section 3.1.1.2, of the receiver.

**receiverGuid**: The *QueryComponentID*, as specified in [MS-SRCHTP] section 3.1.1.2, of the receiver.

**state**: The *State*, as specified in [MS-SRCHTP] section 3.1.1.2, of the receiver.

**upToDate**: A *Boolean* value that is "true" if, and only if, the receiver does not have any outstanding component activity, as specified in [MS-SQLPGAT2] section 3.1.1.35, to do.

**hung**: A *Boolean* value that is "true" if, and only if, the receiver has failed to complete at least one component addition propagation task for more than five minutes.

### 3.4.2 Timers

None.
3.4.3 Initialization
None.

3.4.4 Higher-Layer Triggered Events
None.

3.4.5 Message Processing Events and Sequencing Rules
The following actions MUST be taken by the admin server on a recurring basis, starting with section 3.4.5.1 and ending with section 3.4.5.12. The recurrences SHOULD be one minute apart, but all finite time intervals are allowed by the protocol.

3.4.5.1 Sending a proc_MSS_PropagationGetTasks Message
The admin server calls the proc_MSS_PropagationGetTasks stored procedure, as specified in section 3.1.5.13, with no parameters.

3.4.5.2 Receiving a Propagation All Tasks Result Set
The admin server receives this result set as a result of a call to proc_MSS_PropagationGetTasks, as specified in section 3.4.5.1. When it receives this result set, it MUST replace its current list of running tasks, as specified in section 3.4.1.1, with a new list that contains one running task for each result in this result set, with the following settings:

- **senderID**, as specified in section 3.4.1.1, is set to the value of **SenderID**, as specified in section 3.4.5.2, of the result.
- **catalogID**, as specified in section 3.4.1.1, is set to the value of **CatalogID**, as specified in section 3.4.5.2, of the result.
- **taskType**, as specified in section 3.4.1.1, is set to the value of **TaskType**, as specified in section 3.4.5.2, of the result.
- **objectID**, as specified in section 3.4.1.1, is set to the value of **ObjectID**, as specified in section 3.4.5.2, of the result.
- **maxWorkID**, as specified in section 3.4.1.1, is set to the value of **MaxWorkID**, as specified in section 3.4.5.2, of the result.
- **birthdate**, as specified in section 3.4.1.1, is the value of **BirthDate**, as specified in section 3.4.5.2, of the result.
- **time**, as specified in section 3.4.1.1, is set to the value of **Time**, as specified in section 3.4.5.2, of the result.

3.4.5.3 Sending a proc_MSS_PropagationGetTaskCompletions Message
The admin server calls the proc_MSS_PropagationGetTaskCompletions stored procedure, as specified in section 3.1.5.14, with no parameters.

3.4.5.4 Receiving a Propagation Task Completions Result Set
The admin server receives this result set as a result of a call to proc_MSS_PropagationGetTaskCompletions, as specified in section 3.4.5.3. When it receives
this result set, it MUST replace its current list of task completions, as specified in section 3.4.1.2, with a new list that contains one task completion for each result in this result set, with the following settings:

- **receiverID**, as specified in section 3.4.1.2, is set to the value of **ReceiverID**, as specified in section 3.1.5.14.1, of the result.

- **senderID**, as specified in section 3.4.1.2, is set to the value of **SenderID**, as specified in section 3.1.5.14.1, of the result.

- **catalogID**, as specified in section 3.4.1.2, is set to the value of **CatalogID**, as specified in section 3.1.5.14.1, of the result.

- **taskType**, as specified in section 3.4.1.2, is set to the value of **TaskType**, as specified in section 3.1.5.14.1, of the result.

- **objectId**, as specified in section 3.4.1.2, is set to the value of **ObjectId**, as specified in section 3.1.5.14.1, of the result.

- **maxWorkID**, as specified in section 3.4.1.2, is set to the value of **MaxWorkID**, as specified in section 3.1.5.14.1, of the result.

- **birthdate**, as specified in section 3.4.1.2, is the value of **BirthDate**, as specified in section 3.1.5.14.1, of the result.

### 3.4.5.5 Sending a proc_MSS_PropagationGetErrors Message

The admin server MUST call the **proc_MSS_PropagationGetErrors** stored procedure, as specified in section 3.1.5.12, with no parameters.

### 3.4.5.6 Receiving a Propagation Errors Result Set

The admin server receives this result set as a result of a call to **proc_MSS_PropagationGetErrors**, as specified in section 3.4.5.5. When it receives this result set, it MUST replace its current list of propagation errors, as specified in section 3.4.1.3, with a new list that contains one propagation error for each result in this result set, with the following settings:

- **senderID**, as specified in section 3.4.1.3, is set to the value of **SenderID**, as specified in section 3.1.5.12.1, of the result.

- **receiverID**, as specified in section 3.4.1.3, is set to the value of **ReceiverID**, as specified in section 3.1.5.12.1, of the result.

- **catalogID**, as specified in section 3.4.1.3, is set to the value of **CatalogID**, as specified in section 3.1.5.12.1, of the result.

- **type**, as specified in section 3.4.1.2, is set to the value of **Type**, as specified in section 3.1.5.12.1, of the result.

- **firstUtcTime**, as specified in section 3.4.1.3, is set to the value of **FirstUtcTime**, as specified in section 3.1.5.12.1, of the result.

- **latestUtcTime**, as specified in section 3.4.1.3, is set to the value of **LatestUtcTime**, as specified in section 3.1.5.12.1, of the result.
3.4.5.7 Sending a proc_MSS_GetQueryComponents Message

The admin server calls the proc_MSS_GetQueryComponents stored procedure, as specified in [MS-SRCHTP] section 3.1.5.51, with no parameters.

3.4.5.8 Receiving a Query Components Result Set Message

The admin server receives this result set, as specified in [MS-SRCHTP] section 2.2.4.2, as a result of a call to proc_MSS_GetQueryComponents, as specified in section 3.4.5.7. When it receives this result set, it MUST replace its current list of receivers, as specified in section 3.4.1.4, with a new list that contains one receiver for each result in this result set whose State value, as specified in [MS-SRCHTP] section 2.2.4.2, is either "Ready" or "IndexSplitDone", as specified in [MS-SRCHTP] section 2.2.1.3, with the following settings:

- receiverID, as specified in section 3.4.1.4, is set to the QueryComponentNumber value, as specified in [MS-SRCHTP] section 2.2.4.2, of the result.
- receiverGuid, as specified in section 3.4.1.4, is set to the QueryComponentID value, as specified in [MS-SRCHTP] section 2.2.4.2, of the result.
- upToDate, as specified in section 3.4.1.4, is set to "true".
- hung, as specified in section 3.4.1.4, is set to "false".

3.4.5.9 Sending a proc_MSS_GetComponentStatusUpToDate Message

The admin server MUST call the proc_MSS_GetComponentStatusUpToDate stored procedure, as specified in [MS-SQLPGAT2] section 3.1.5.18, twice for each receiver in the list of receivers, as specified in section 3.4.1.4.

The first call MUST be made with the following parameters:

- @ComponentType, as specified in [MS-SQLPGAT2] section 3.1.5.18, MUST be "1".
- @ReceiverID, as specified in [MS-SQLPGAT2] section 3.1.5.18. MUST be the value of receiverID, as specified in section 3.4.1.4.
- @ProjectName, as specified in [MS-SQLPGAT2] section 3.1.5.18, MUST be "Portal_Content".

If the value of the output parameter @UpToDate is "false", the upToDate value of the receiver MUST be set to "false".

The second call MUST be made with the following parameters:

- @ComponentType, as specified in [MS-SQLPGAT2] section 3.1.5.18, MUST be "1".
- @ReceiverID, as specified in [MS-SQLPGAT2] section 3.1.5.18, MUST be the value of receiverID, as specified in section 3.4.1.4.
- @ProjectName, as specified in [MS-SQLPGAT2] section 3.1.5.18, MUST be "AnchorProject".

If the value of the output parameter @UpToDate is "false", the upToDate value, as specified in section 3.4.1.4, of the receiver MUST be set to "false".
3.4.5.10 Sending a proc_MSS_PropagationReportError Message

For each full-text index catalog Portal_Content and AnchorProject, as specified in [MS-CIFO], of each receiver in the list of receivers, as specified in section 3.4.1.4, the admin server MUST call the proc_MSS_PropagationReportError stored procedure, as specified in section 3.1.5.9, in the following cases.

If the receiver's value of hung, as specified in section 3.4.1.4, is "true", and there is no propagation error in the list of propagation errors, as specified in section 3.4.1.3, in which the receiverID value, as specified in section 3.4.1.3, is the receiverID, as specified in section 3.4.1.4, of the receiver, catalogID value, as specified in section 3.4.1.3, is the catalog identifier, as specified in section 2.2.1.2, of the full-text index catalog, type value, as specified in section 3.4.1.3, is "ReceiverHang", as specified in section 2.2.1.3, and latestUtcTime value, as specified in section 3.4.1.3, is less than five minutes before the current time, the admin server MUST call the proc_MSS_PropagationReportError stored procedure, as specified in section 3.1.5.9, with the following parameters:

- @SenderID, as specified in section 3.1.5.9, MUST be NULL.
- @ReceiverID, as specified in section 3.1.5.9, MUST be the value of receiverID, as specified in section 3.4.1.4.
- @CatalogID, as specified in section 3.1.5.9, MUST be the catalog identifier, as specified in section 2.2.1.2, corresponding to the full-text index catalog.
- @Type, as specified in section 3.1.5.9, MUST be "ReceiverHang", as specified in section 2.2.1.3.
- @Message, as specified in section 3.1.5.9, SHOULD be "not completing tasks", but any text is allowed by this protocol.

If the requirements to make the first call in the preceding description were not met, the upToDate value, as specified in section 3.4.1.4, of the receiver is "false", and there is no propagation error in the list of propagation errors, as specified in section 3.4.1.3, in which the receiverID value, as specified in section 3.4.1.3, is the receiverID, as specified in section 3.4.1.4, of the receiver, catalogID value, as specified in section 3.4.1.3, is the catalog identifier, as specified in section 2.2.1.2, of the full-text index catalog, type value, as specified in section 3.4.1.3, is "ReceiverHang", as specified in section 2.2.1.3, and latestUtcTime value, as specified in section 3.4.1.3, is less than five minutes before the current time, the admin server MUST call the proc_MSS_PropagationReportError stored procedure, as specified in section 3.1.5.9, with the following parameters:

- @SenderID, as specified in section 3.1.5.9, MUST be NULL.
- @ReceiverID, as specified in section 3.1.5.9, MUST be the value of receiverID, as specified in section 3.4.1.4.
- @CatalogID, as specified in section 3.1.5.9, MUST be the catalog identifier, as specified in section 2.2.1.2, corresponding to the full-text index catalog.
- @Type, as specified in section 3.1.5.9, MUST be "ReceiverHang", as specified in section 2.2.1.3.
- @Message, as specified in section 3.1.5.9, SHOULD be "not pausing, resuming, or resetting", but any text is allowed by this protocol.
### 3.4.5.11 Sending a proc_MSS_PropagationDeleteError Message

For each full-text index catalog **Portal_Content** and **AnchorProject**, as specified in [MS-CIFO], of each receiver in the list of receivers, as specified in section 3.4.1.4, the admin server MUST call the proc_MSS_PropagationDeleteError stored procedure, as specified in section 3.1.5.10, for the following case.

If the **upToDate** value, as specified in section 3.4.1.4, of the receiver is "true", and the value of **hung**, as specified in section 3.4.1.4, is "false", and there is a propagation error in the list of propagation errors, as specified in section 3.4.1.3, whose receiverID value, as specified in section 3.4.1.3, is the receiver identifier, as specified in section 3.4.1.4, of the receiver, catalogID value, as specified in section 3.4.1.3, is the catalog identifier, as specified in section 2.2.1.2, of the full-text index catalog, and type value, as specified in section 3.4.1.3, is "ReceiverHang", as specified in section 2.2.1.3, the admin server MUST call the proc_MSS_PropagationReportError stored procedure, as specified in section 3.1.5.10, with the following parameters:

- **@SenderID**, as specified in section 3.1.5.10, MUST be NULL.
- **@ReceiverID**, as specified in section 3.1.5.10, MUST be the receiver identifier, as specified in section 3.4.1.4.
- **@CatalogID**, as specified in section 3.1.5.10 MUST be the catalog identifier, as specified in section 2.2.1.2, corresponding to the full-text index catalog.
- **@Type**, as specified in section 3.1.5.10, MUST be "ReceiverHang".

### 3.4.5.12 Sending a proc_MSS_SetQueryComponent Message

For each receiver in the list of receivers, as specified in section 3.4.1.4, for which there is at least one propagation error in the list of propagation errors, as specified in section 3.4.1.3, where receiver identifier, as specified in section 3.4.1.4, and firstUtcTime, as specified in section 3.4.1.3, is more than one hour before the current time, the admin server MUST call the proc_MSS_SetQueryComponent stored procedure, as specified in [MS-SRCHTP] section 3.1.5.87, with the following parameters:

- **@QueryComponentID** MUST be the **receiverGuid**, as specified in section 3.4.1.4, of the receiver.
- **@State** MUST be "Offline", as specified in [MS-SRCHTP] section 2.2.1.3.
- All other parameters MUST be NULL.

### 3.4.6 Timer Events

None.

### 3.4.7 Other Local Events

None.
4 Protocol Examples

4.1 Component Addition Propagation

For the example in the following subsections, a search application is demonstrated that has the following four actors:

- **DB-1**: a back-end database server.
- **SEN-1**: a sender.
- **REC-1**: a query component.
- **REC-2**: another query component.

4.1.1 Initial State

4.1.1.1 DB-1

4.1.1.1.1 List of Ready Query Components

The list contains the following two query components:

1. REC-1
   1. **QueryComponentNumber** is zero.
   2. **ServerName** is "REC-1".
   3. **State** is "Ready".
   4. **ShareName** is "4c436ee0-b809-4e8a-b00b-be776306e0ee-query-0".

2. REC-2
   1. **QueryComponentNumber** is "1".
   2. **ServerName** is "REC-2".
   3. **State** is "Ready".
   4. **ShareName** is "4c436ee0-b809-4e8a-b00b-be776306e0ee-query-1".

4.1.1.1.2 List of Running Tasks

The list is empty.

4.1.1.2 SEN-1

4.1.1.2.1 Search Application Name

The name is "4c436ee0-b809-4e8a-b00b-be776306e0ee".

4.1.1.2.2 Sender Identifier

The sender identifier is zero.
4.1.1.2.3 List of Ready Query Components

The list contains the following two query components:

1. REC-1
   1. serverName is "REC-1".
   2. shareName is "4c436ee0-b809-4e8a-b00b-be776306e0ee-query-0".

2. REC-2
   1. serverName is "REC-2".
   2. shareName is "4c436ee0-b809-4e8a-b00b-be776306e0ee-query-1".

4.1.1.2.4 List of Completed Tasks

The list is empty.

4.1.1.3 REC-1

4.1.1.3.1 Receiver Identifier

The receiver identifier is zero.

4.1.1.3.2 List of Incomplete Tasks

The list is empty.

4.1.1.4 REC-2

4.1.1.4.1 Receiver Identifier

The receiver identifier is "1".

4.1.1.4.2 List of Incomplete Tasks

The list is empty.

4.1.2 Sequence of Events

Events 1 through 6 are not necessary for the propagation sequence to occur, but are presented to demonstrate the steady state of the system that would be recurring in cycles before the propagation sequence begins in event 7.

Event 1: SEN-1 polls DB-1 every 30 seconds by calling the proc_MSS_PropagationIndexerGetCompletedTasks stored procedure with @CatalogID set to "1". DB-1 returns zero.

Event 2: DB-1 replies with an empty completed tasks result set, indicating that there are no completed tasks for the main catalog.

Event 3: REC-1 polls DB-1 every 30 seconds by calling the proc_MSS_PropagationQueryComponentPickUpNewPropagationItems stored procedure with @CatalogID set to "1". DB-1 returns zero.
**Event 4:** DB-1 replies with an empty incomplete tasks result set, indicating that there are currently no propagation tasks for REC-1 to perform.

**Event 5:** REC-2 polls DB-1 every 30 seconds by calling the `proc_MSS_PropagationQueryComponentPickUpNewPropagationItems` stored procedure with `@CatalogID` set to zero. DB-1 returns zero.

**Event 6:** DB-1 replies with an empty incomplete tasks result set, indicating that there are currently no propagation tasks for REC-2 to perform.

At this point, the sender has generated a new full-text index component and propagates the component. The full-text index component has index identifier "0x0001001A", versioned index identifier "0x0054001A", maximum document identifier (1) "471952", and component birth date "414".

**Event 7:** SEN-1 writes the full-text index component files listed in the following table. All file names begin with the sender identifier zero, contain one of the file names of a full-text index component, and end with the "cp" extension.

<table>
<thead>
<tr>
<th>File name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000.0001001A.ci.cp</td>
</tr>
<tr>
<td>0000.0001001A.dir.cp</td>
</tr>
<tr>
<td>0000.0001001A.bsi.cp</td>
</tr>
<tr>
<td>0000.0001001A.bsd.cp</td>
</tr>
<tr>
<td>0000.0001001A.csi.cp</td>
</tr>
<tr>
<td>0000.0001001A.csd.cp</td>
</tr>
<tr>
<td>0000.0001001A.wid.cp</td>
</tr>
<tr>
<td>0000.0001001A.list.cp</td>
</tr>
</tbody>
</table>

to both of the following file paths:

**Shared folder**

- `\\REC-1\4c436ee0-b809-4e8a-b00b-be776306e0ee-query-0\4c436ee0-b809-4e8a-b00b-be776306e0ee-query-0\Projects\Portal_Content\Indexer\Cifiles`
- `\\REC-2\4c436ee0-b809-4e8a-b00b-be776306e0ee-query-1\4c436ee0-b809-4e8a-b00b-be776306e0ee-query-1\Projects\Portal_Content\Indexer\Cifiles`

**Event 8:** SEN-1 calls the `proc_MSS_PropagationIndexerInsertNewTask` stored procedure with the following parameters:

- `@SenderID` is set to zero.
- `@CatalogID` is set to "1".
- `@TaskType` is set to "ComponentAddition".
• @ObjectID is set to "5505050" (hexadecimal equivalent: 0x0054001A).

• @MaxWorkID is set to "471952".

• @BirthDate is set to "414".

DB-1 returns zero.

**Event 9:** REC-1 calls the proc_MSS_PropagationQueryComponentPickUpNewPropagationItems stored procedure with @ReceiverID set to zero and @CatalogID set to "1". DB-1 returns zero.

**Event 10:** DB-1 sends a propagation tasks result set with one result with the following settings:

• SenderID is set to zero.

• CatalogID is set to "1".

• TaskType is set to "ComponentAddition".

• ObjectID is set to "5505050" (hexadecimal equivalent: 0x0054001A).

• MaxWorkID is set to "471952".

• Birthdate is set to "414".

**Event 11:** REC-1 applies the full-text index component and calls the proc_MSS_PropagationQueryComponentReportTaskReady stored procedure with the following parameters:

• @ReceiverID is set to zero.

• @SenderID is set to zero.

• @CatalogID is set to "1".

• @TaskType is set to "ComponentAddition".

• @ObjectID is set to "5505050".

DB-1 returns zero.

**Event 12:** SEN-1 polls DB-1 again by calling the proc_MSS_PropagationIndexerGetCompletedTasks stored procedure with @CatalogID set to "1". DB-1 returns zero.

**Event 13:** DB-1 sends an empty completed tasks result set, indicating that there are no completed tasks for the main catalog.

**Event 14:** REC-2 calls the proc_MSS_PropagationQueryComponentPickUpNewPropagationItems stored procedure with @ReceiverID set to "1" and @CatalogID set to "1".

**Event 15:** DB-1 returns the following propagation tasks result set:

• SenderID is set to zero.

• CatalogID is set to "1".
- **TaskType** is set to "ComponentAddition".
- **ObjectID** is set to "5505050".
- **MaxWorkID** is set to "471952".
- **Birthdate** is set to "414".

**Event 16:** REC-2 applies the full-text index component and calls the `proc_MSS_PropagationQueryComponentReportTaskReady` stored procedure with the following parameters:
- @ReceiverID is set to "1".
- @SenderID is set to "1".
- @CatalogID is set to "1".
- @TaskType is set to "ComponentAddition".
- @ObjectID is set to "5505050".

DB-1 returns zero.

**Event 17:** The index server SEN-1 polls DB-1 again by calling the `proc_MSS_PropagationIndexerGetCompletedTasks` stored procedure with @SenderID set to zero and @CatalogID set to "1". DB-1 returns zero.

**Event 18:** DB-1 returns the following completed tasks result set:
- SenderID is set to zero.
- CatalogID is set to "1".
- TaskType is set to "ComponentAddition".
- ObjectID is set to "5505050".
- {MaxWorkID} is set to zero.
- {Birthdate} is set to zero.

**Event 19:** The index server SEN-1 then calls the `proc_MSS_PropagationIndexerCleanUpTablesForTask` stored procedure with the following parameters:
- @SenderID is set to zero.
- @CatalogID is set to "1".
- @TaskType is set to "ComponentAddition".
- @ObjectID is set to "5505050".

DB-1 returns zero.

**Event 20:** DB-1 deletes the propagation task from its list of running propagation tasks.
5 Security

5.1 Security Considerations for Implementers

Security for this protocol is controlled by the access rights to the databases on the back-end database server, which is negotiated as part of the TDS protocol, as described in [MS-TDS].

To call stored procedures, the sender and receiver runs as an account that has read and write permissions on the back-end database server. That account is a member of the local security group (2) named "WSS_WPG".

5.2 Index of Security Parameters

None.
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:

- Microsoft® SharePoint® Server 2010
- Microsoft® SQL Server® 2005
- Microsoft® SQL Server® 2008
- Microsoft® SQL Server® 2008 R2

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.
7 Change Tracking

No table of changes is available. The document is either new or has had no changes since its last release.
8 Index

A

Abstract data model
admin server 41
back-end database server 16
receiver 37
sender 30
server (section 3.1.1 16, section 3.4.1 41)
Admin server
abstract data model 41
higher layer triggered events 43
initialization 43
list of propagation errors 42
list of running tasks 41
message processing events 43
other local events 47
overview 41
receiving a propagation all tasks result set (section 3.4.5.2 43, section 3.4.5.6 44)
receiving a propagation task completions result set 43
receiving a query components result set message 45
sending a proc_MSS_GetComponentStatusUpToDate message 45
sending a proc_MSS_GetQueryComponents message 45
sending a proc_MSS_PropagationDeleteError message 47
sending a proc_MSS_PropagationGetErrors message 44
sending a proc_MSS_PropagationGetTaskCompletions message 43
sending a proc_MSS_PropagationGetTasks message 43
sending a proc_MSS_PropagationReportError message 47
sequencing rules 43
timer events 47		timers 42
Admin server interface 41
Applicability 11
Attribute groups - overview 15
Attributes - overview 15

B

Back-End database server
abstract data model 16
higher layer triggered events 18
initialization 17
interface 16
list of propagation errors 17
list of ready query components 16
list of running tasks 16
message processing events 18
other local events 30
overview 16
sequencing rules 18
timer events 30
timers 17
Back-End database server - methods
proc_MSS_PropagationDeleteError 26
proc_MSS_PropagationDeleteErrors 27
proc_MSS_PropagationGetErrors 28
proc_MSS_PropagationGetTaskCompletions 29
proc_MSS_PropagationGetTasks 28
proc_MSS_PropagationIndexerCleanUpTablesForTask 18
proc_MSS_PropagationIndexerDeleteAllTasksFromSender 26
proc_MSS_PropagationIndexerGetCompletedTask $ 19
proc_MSS_PropagationIndexerGetReadyQueryComponents 20
proc_MSS_PropagationIndexerGetTasks 20
proc_MSS_PropagationQueryComponentPickUpNewPropagationItems 22
proc_MSS_PropagationQueryComponentReportTaskReady 24
proc_MSS_PropagationReportError 26
Back-End database server - result sets
proc_MSS_PropagationGetErrors – propagation errors 28
proc_MSS_PropagationGetTaskCompletions – propagation task completions 29
proc_MSS_PropagationGetTasks – propagation all tasks 29
proc_MSS_PropagationIndexerGetCompletedTask $ - completed tasks 19
proc_MSS_PropagationIndexerGetReadyQueryComponents – ready query components 20
proc_MSS_PropagationIndexerGetTasks – propagation tasks 21
proc_MSS_PropagationQueryComponentPickUpNewPropagationItems – propagation tasks 23
Back-end database server interface 16
Binary structures
Full-Text Index Component Message 13
Versioned Index Identifier 14
Binary structures - overview 13
Bit fields - overview 13

C

Capability negotiation 11
Catalog ID simple type 12
Change tracking 55
Client
overview 16
Common data types
overview 12
Completed Tasks result set
     proc_MSS_PropagationIndexerGetCompletedTasks 19
Complex types - overview 14
Component Addition Propagation example 48

D
Data model - abstract
     admin server 41
     back-end database server 16
     receiver 37
     sender 30
server (section 3.1.1, section 3.4.1)
Data types
     Catalog ID simple type 12
     common 12
     Propagation Error Type simple type 12
     Task Type simple type 12
Data types - simple
     Catalog ID 12
     overview 12
     Propagation Error Type 12
     Task Type 12

E
Elements - overview 14
     Error possibly exists 31
     Error possibly exists - receiver 38
Events
     local - server (section 3.1.7, section 3.4.7)
     timer - server (section 3.1.6, section 3.4.6)
Example
     Component Addition Propagation 48
     sequence of events 49

F
Fields - vendor-extensible 11
Flag structures - overview 13
Full-Text Index Component Message binary structure 13

G
Glossary 7
Groups - overview 15

H
Higher layer triggered events
     Admin server 43
     Back-End database server 18
     receiver 38
     sender 31
Higher-layer triggered events
server (section 3.1.4, section 3.4.4)

I
ID
     receiver 37
     sender 30
Implementer - security considerations 53
Index of security parameters 53
Informative references 8
Initialization
     admin server 43
     back-end database server 17
     receiver 38
     sender 31
server (section 3.1.3, section 3.4.3)
Interfaces - server
     admin 41
     back-end database 16
Introduction 7

L
List of completed tasks 31
List of incomplete tasks 38
List of propagation errors
     admin server 42
     back-end database server 17
List of ready query components (section 3.1.1, section 3.2.1.3, section 3.4.4)
List of receivers 42
List of running tasks
     admin server 41
     back-end database server 16
     sender 31
List of task completions 41
Local events
server (section 3.1.7, section 3.4.7)

M
Message processing
server (section 3.1.5, section 3.4.5)
Message processing events
     admin server 43
     back-end database server 18
     receiver 38
     sender 32
Messages
     attribute groups 15
     attributes 15
     binary structures 13
     bit fields 13
     common data types 12
     complex types 14
     elements 14
     enumerations 12
     flag structures 13
     Full-Text Index Component Message binary structure 13
     groups 15
     namespaces 14
     result sets 14

[MS-CIPROP2] — v20120630
Index Propagation Version 2 Protocol Specification

Copyright © 2012 Microsoft Corporation.

Release: July 16, 2012
Method

- **proc_MSS_PropagationDeleteError** method 26
- **proc_MSS_PropagationDeleteErrors** method 27
- **proc_MSS_PropagationGetErrors** method 28
- **proc_MSS_PropagationGetTaskCompletions** method 29
- **proc_MSS_PropagationGetTasks** method 28
- **proc_MSS_PropagationIndexerCleanUpTablesForTask** method 18
- **proc_MSS_PropagationIndexerDeleteAllTasksFromSender** method 24
- **proc_MSS_PropagationIndexerGetCompletedTasks** method 19
- **proc_MSS_PropagationIndexerGetReadyQueryComponents** method 20
- **proc_MSS_PropagationIndexerGetTasks** method 20
- **proc_MSS_PropagationIndexerInsertNewTask** method 21
- **proc_MSS_PropagationQueryComponentPickUpNewPropagationItems** method 22
- **proc_MSS_PropagationQueryComponentReportTaskReady** method 24
- **proc_MSS_PropagationReportError** method 26

Messages

- **Receiving a Propagation All Tasks Result Set** 43
- **Receiving a Propagation Errors Result Set** 44
- **Receiving a Propagation Task Completions Result Set** 43
- **Receiving a Query Components Result Set Message** 45
- **Sending a proc_MSS_GetComponentStatusUpToDate Message** 45
- **Sending a proc_MSS_GetQueryComponents Message** 45
- **Sending a proc_MSS_PropagationDeleteError Message** 47
- **Sending a proc_MSS_PropagationGetErrors Message** 44
- **Sending a proc_MSS_PropagationGetTaskCompletions Message** 43
- **Sending a proc_MSS_PropagationGetTasks Message** 43
- **Sending a proc_MSS_PropagationReportError Message** 46
- **Sending a proc_MSS_SetQueryComponent Message** 47

Namespaces

- **Namespaces** 14
- **Normative references** 8

Other local events

- **Other local events**
  - **admin server** 47

Parameters - security index

- **Parameters - security index** 53
- **Preconditions** 11
- **Prerequisites** 11
  - **proc_MSS_PropagationDeleteError method** 26
  - **proc_MSS_PropagationDeleteErrors method** 27
  - **proc_MSS_PropagationGetErrors method** 28
  - **proc_MSS_PropagationGetTaskCompletions method** 29
  - **proc_MSS_PropagationGetTasks method** 28
  - **proc_MSS_PropagationIndexerCleanUpTablesForTask method** 18
  - **proc_MSS_PropagationIndexerDeleteAllTasksFromSender method** 24
  - **proc_MSS_PropagationIndexerGetCompletedTasks method** 19
  - **proc_MSS_PropagationIndexerGetReadyQueryComponents method** 20
  - **proc_MSS_PropagationIndexerInsertNewTask method** 21
  - **proc_MSS_PropagationQueryComponentPickUpNewPropagationItems method** 22
  - **proc_MSS_PropagationQueryComponentReportTaskReady method** 24
  - **proc_MSS_PropagationReportError method** 26

Product behavior

- **Product behavior** 54
- **Propagation All Tasks result set**
  - **proc_MSS_PropagationGetTasks** 29
- **Propagation Error Type simple type** 12
- **Propagation Errors result set**
  - **proc_MSS_PropagationGetErrors** 28
- **Propagation Task Completions result set**
  - **proc_MSS_PropagationGetTaskCompletions** 29
- **Propagation Tasks result set**
  - **proc_MSS_PropagationIndexerGetTasks** 21
  - **proc_MSS_PropagationQueryComponentPickUpNewPropagationItems** 23

Receiver

- **abstract data model** 37
- **error possibly exists** 38
- **higher layer trigged events** 38
- **ID** 37
- **initialization** 38
- **list of incomplete tasks** 38
- **message processing events** 38
- **other local events** 41
- **overview** 37
- **receiving a propagation tasks result set** 39
sending a proc_MSS_PropagationDeleteError message 40
sending a proc_MSS_PropagationQueryComponentPickUp NewPropagationItems message 38
sending a proc_MSS_PropagationQueryComponentReport TaskReady message 40
sending a proc_MSS_PropagationReportError message 39
sequencing rules 38
timer events 41
timers 38
Receiving a Propagation All Tasks Result Set method 43
Receiving a Propagation Errors Result Set method 44
Receiving a Propagation Task Completions Result Set method 43
Receiving a Query Components Result Set Message method 45
References
informative 8
normative 8
Relationship to other protocols 10
Result sets
proc_MSS_PropagationGetErrors – propagation errors 28
proc_MSS_PropagationGetTaskCompletions – propagation task completions 29
proc_MSS_PropagationGetTasks – propagation all tasks 29
proc_MSS_PropagationIndexerGetCompletedTasks – completed tasks 19
proc_MSS_PropagationIndexerGetReadyQueryComponents – ready query components 20
proc_MSS_PropagationIndexerGetTasks – propagation tasks 21
proc_MSS_PropagationQueryComponentPickUpNewPropagationItems – propagation tasks 23
Result sets - overview 14
S
Search application name 30
Security
implementer considerations 53
parameter index 53
Sender
abstract data model 30
error possibly exists 31
higher layer triggered events 31
ID 30
initialization 31
list of completed tasks 31
list of ready query components 30
list of running tasks 31
message processing events 32
other local events 37
overview 30
receiving a completed tasks result set message 36
receiving a propagation tasks result set 37
receiving a ready query components result set 32
search application name 30
sending a full-text index component message 32
sending a proc_MSS_PropagationDeleteError message 34
sending a proc_MSS_PropagationIndexerCleanUpTablesForTask message 36
sending a proc_MSS_PropagationIndexerDeleteAllTasksFromSender message 37
sending a proc_MSS_PropagationIndexerGetCompletedTasks message 35
sending a proc_MSS_PropagationIndexerGetReadyQueryComponents message 32
sending a proc_MSS_PropagationIndexerGetTasks message 36
sending a proc_MSS_PropagationIndexerInsertNewTask message 34
sending a proc_MSS_PropagationReportError message 33
sequencing rules 32
timer events 37
timers 37
Sending a proc_MSS_GetComponentStatusUpToDate Message method 45
Sending a proc_MSS_GetQueryComponents Message method 45
Sending a proc_MSS_PropagationDeleteError Message method 47
Sending a proc_MSS_PropagationGetErrors Message method 44
Sending a proc_MSS_PropagationGetTaskCompletions Message method 43
Sending a proc_MSS_PropagationGetTasks Message method 43
Sending a proc_MSS_PropagationReportError Message method 46
Sending a proc_MSS_SetQueryComponent Message method 47
Sequencing rules
admin server 43
back-end database server 18
receiver 38
sender 32
server (section 3.1.5, section 3.4.5 43)
Server
abstract data model (section 3.1.1, section 3.4.1 41)
higher-layer triggered events (section 3.1.4, section 3.4.4 43)
initialization (section 3.1.3, section 3.4.3 43)
local events (section 3.1.7, section 3.4.7 47)
message processing (section 3.1.5 18, section 3.4.5 43)
overview 16
proc_MSS_PropagationDeleteError method 26
proc_MSS_PropagationDeleteErrors method 27
proc_MSS_PropagationGetErrors method 28
proc_MSS_PropagationGetTaskCompletions method 29
proc_MSS_PropagationGetTasks method 28
proc_MSS_PropagationIndexerCleanUpTablesForTask method 18
proc_MSS_PropagationIndexerDeleteAllTasksFromSender method 24
proc_MSS_PropagationIndexerGetCompletedTasks method 19
proc_MSS_PropagationIndexerGetReadyQueryComponents method 20
proc_MSS_PropagationIndexerGetTasks method 20
proc_MSS_PropagationIndexerInsertNewTask method 21
proc_MSS_PropagationQueryComponentPickUpNewPropagationItems method 22
proc_MSS_PropagationQueryComponentReportTaskReady method 24
proc_MSS_PropagationReportError method 26
Receiving a Propagation All Tasks Result Set method 43
Receiving a Propagation Errors Result Set method 44
Receiving a Propagation Task Completions Result Set method 43
Receiving a Query Components Result Set Message method 45
Sending a proc_MSS_GetComponentStatusUpToDate Message method 45
Sending a proc_MSS_GetQueryComponents Message method 45
Sending a proc_MSS_PropagationDeleteError Message method 47
Sending a proc_MSS_PropagationGetErrors Message method 44
Sending a proc_MSS_PropagationGetTaskCompletions Message method 43
Sending a proc_MSS_PropagationGetTasks Message method 43
Sending a proc_MSS_PropagationReportError Message method 46
Sending a proc_MSS_SetQueryComponent Message method 47
sequencing rules (section 3.1.5 18, section 3.4.5 43)
timer events (section 3.1.6 30, section 3.4.6 47)
timers (section 3.1.2 17, section 3.4.2 42)
Simple data types
Catalog ID 12
overview 12
Propagation Error Type 12
Task Type 12
Simple types - overview 14
Standards assignments 11
Structures
binary 13
table and view 14
XML 14
T
Table structures - overview 14
Task Type simple type 12
Timer events
admin server 47
back-end database server 30
receiver 41
sender 37
server (section 3.1.6 30, section 3.4.6 47)
Timers
admin server 42
back-end database server 17
receiver 38
sender 31
server (section 3.1.2 17, section 3.4.2 42)
Tracking changes 55
Transport 12
Triggered events - higher-layer
server (section 3.1.4 18, section 3.4.4 43)
Types
complex 14
simple 14
V
Vendor-extensible fields 11
Versioned Index Identifier binary structure 14
Versioning 11
View structures - overview 14
X
XML structures 14