[MS-CIPROP]:
Index Propagation Protocol Specification

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's Open Specification Promise (available here: http://www.microsoft.com/interop/osp) or the Community Promise (available here: http://www.microsoft.com/interop/cp/default.mspx). 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

Date	Revision History	Revision Class	Comments
04/04/2008	0.1		Initial Availability
06/27/2008	1.0	Major	Revised and edited the technical content
10/06/2008	1.01	Editorial	Revised and edited the technical content
12/12/2008	1.02	Editorial	Revised and edited the technical content
07/13/2009	1.03	Major	Changes made for template compliance
08/28/2009	1.04	Editorial	Revised and edited the technical content
11/06/2009	1.05	Editorial	Revised and edited the technical content
02/19/2010	2.0	Editorial	Revised and edited the technical content
03/31/2010	2.01	Editorial	Revised and edited the technical content
04/30/2010	2.02	Editorial	Revised and edited the technical content
06/07/2010	2.03	Editorial	Revised and edited the technical content
06/29/2010	2.04	Editorial	Changed language and formatting in the technical content.
07/23/2010	2.04	No change	No changes to the meaning, language, or formatting of the technical content.
09/27/2010	2.04	No change	No changes to the meaning, language, or formatting of the technical content.
11/15/2010	2.04	No change	No changes to the meaning, language, or formatting of the technical content.
12/17/2010	2.04	No change	No changes to the meaning, language, or formatting of the technical content.
03/18/2011	2.04	No change	No changes to the meaning, language, or formatting of the technical content.
06/10/2011	2.04	No change	No changes to the meaning, language, or formatting of the technical content.

Table of Contents

1 Introduction

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

1.1 Glossary

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

little-endian
Unicode

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

back-end database server
component
component birth date
crawl component
datetime
document identifier
document set
farm
full-text index catalog
full-text index component
index identifier
index partition
query
query component
query topology
result set
return code
search service application
stored procedure
task
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

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"

[MSDN-TSQL-Ref] Microsoft Corporation, "Transact-SQL Reference", http://msdn.microsoft.com/en-us/library/ms189826(SQL.90).aspx

[MS-SMB] Microsoft Corporation, "Server Message Block (SMB) Protocol Specification".

[MS-SQL] Microsoft Corporation, "SQL Server 2000 Architecture and XML/Internet Support", Volume 1 of Microsoft SQL Server 2000 Reference Library, Microsoft Press, 2001, ISBN 0-7356-1280-3, http://msdn.microsoft.com/en-us/library/dd631854(v=SQL.10).aspx

[MS-SRCHTP] Microsoft Corporation, "Search Topology Protocol Specification"

[MS-TDS] Microsoft Corporation, "Tabular Data Stream Protocol Specification".

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

1.2.2 Informative References

[MS-GLOS] Microsoft Corporation, "Windows Protocols Master Glossary".

[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 (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 Microsoft Office SharePoint Server Search Application.

This protocol is used to synchronize changes made to a full-text index catalog from either static rank computation or component addition across receivers. Component addition includes newly crawled content, comprising additions, revisions, and removals. Also, propagation of the static rank computation is necessary as exactly same queries may be routed to different receivers each time they are made, therefore, the static rank computation increases the probability of retrieving similar results across multiple searches.

In a nutshell, senders inform the back-end database server of any changes, while receivers regularly poll back-end database server for timely propagation of changes & updates. On the other hand, receivers inform the back-end database server that they are up-to-date & this information is propagated to senders through the back-end database server.

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.

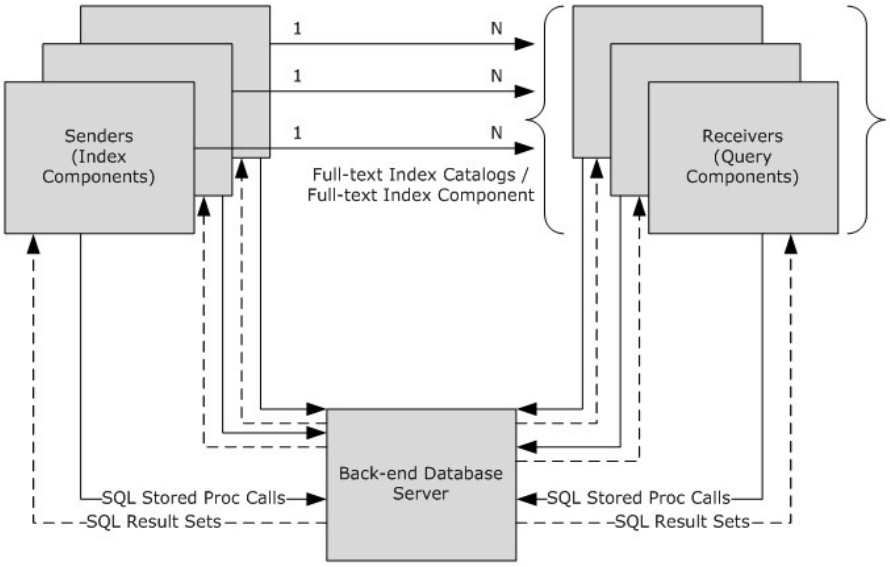


Figure 1: High-level view of communication between servers

The following figure shows the sequence of events during the process of propagation

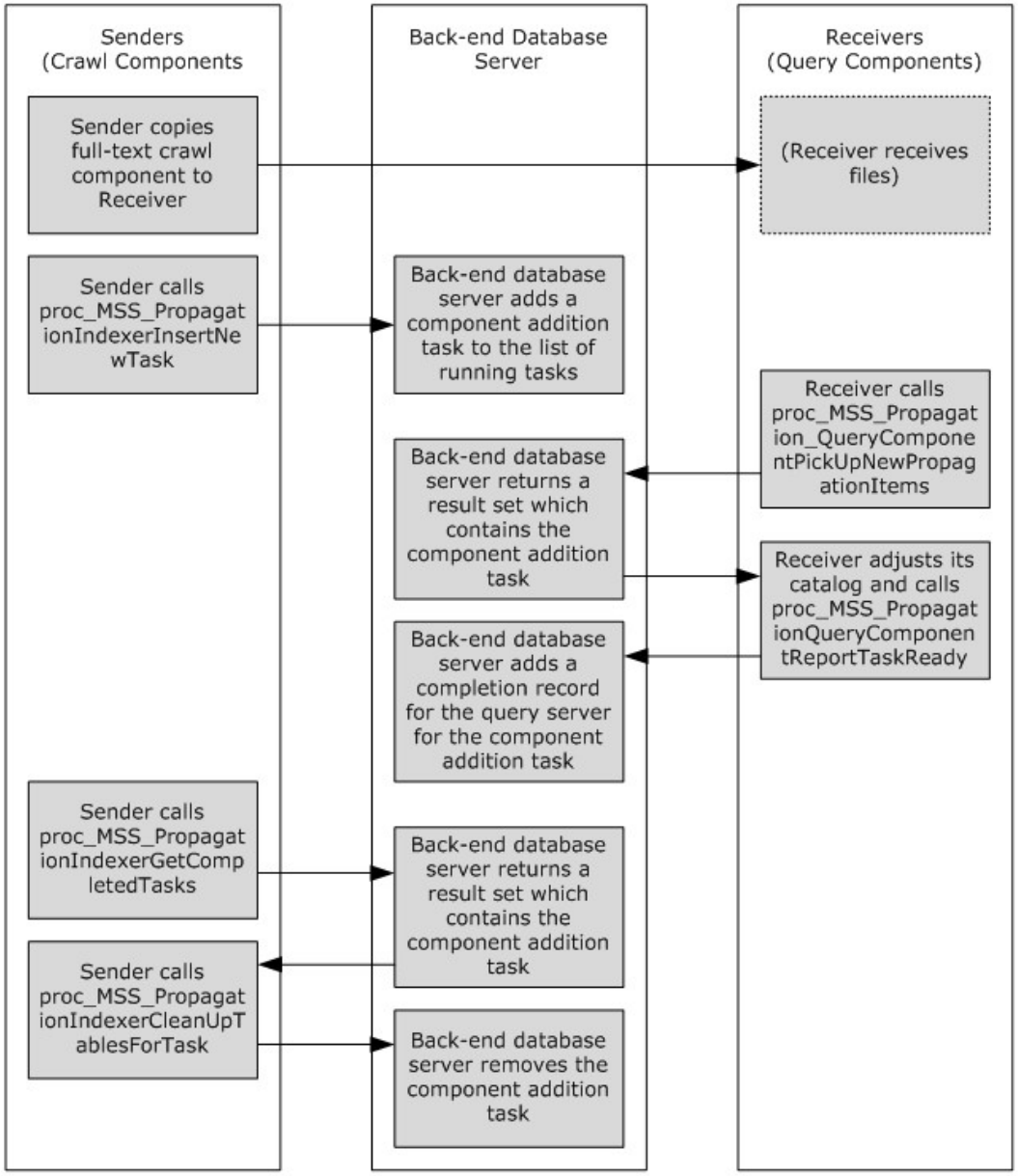


Figure 2: Sequence of operations used to propagate the full-text index component

1.4 Relationship to Other Protocols

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

This protocol relies on Server Message Control Block (SMB) Specification [MS-SMB] as its transport protocol to perform server-to-server file copies.

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:

1. There is a file system share on each query server that allows read and write operation by the account named "WSS_WPG" on that query server.

2. The stored procedures specified in this document are present on the back-end database server.

3. 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** serving a Microsoft Office SharePoint Server search application.

1.7 Versioning and Capability Negotiation

Security and Authentication Methods: This protocol supports the SSPI and SQL Authentication with the Protocol Server role specified in [MS-TDS].

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2 Messages

2.1 Transport

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

2.2 Message Syntax

2.2.1 Common Data Types

2.2.1.1 Simple Data Types and Enumerations

2.2.1.2 Enumerations

2.2.1.2.1 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.

Symbolic name	Value	Description
ComponentAddition	1	A **full-text index component** will be received by each receiver.
StaticRankComputation	2	All activities performed during a static rank computation event will be performed by each receiver.

2.2.1.2.2 CatalogID

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

Value	Description
1	The main catalog, as specified in [MS-CIFO] section 2.18.1.
2	The anchor text catalog, as specified in [MS-CIFO] section 2.18.2.

2.2.2 Full-Text Index Component Message

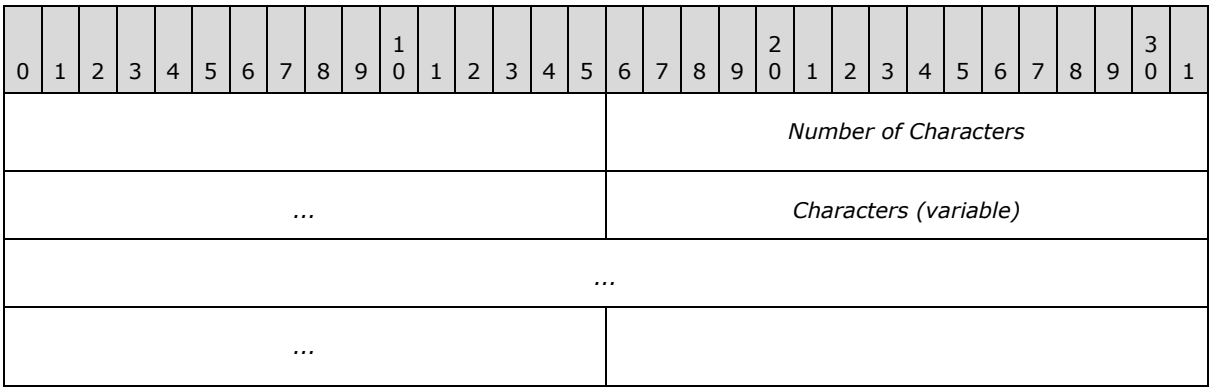
The unit of transfer in full-text index component propagation is a set of files. Each file in this set, except for one, is a duplicate of a file in a full-text index component ([MS-CIFO] section 2.17) in content, but the extension ".cp" is appended to the original file name to create the name of the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

The one file in this set which does not correspond to a full-text index component file is a propagation list file (section 2.2.2.1). See section 4.1 for an example.

2.2.2.1 Propagation List File

The .list file is a list of **Unicode** strings stored in the string list format specified in section 2.2.2.1. All integers and characters are stored in **little-endian** form unless specified otherwise.

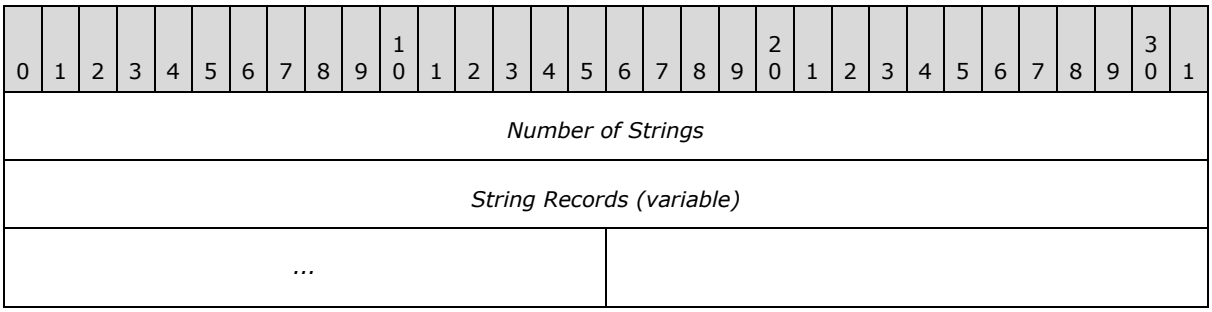
2.2.2.1.1 String Record



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. It MUST terminate at 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.

2.2.2.1.2 String List

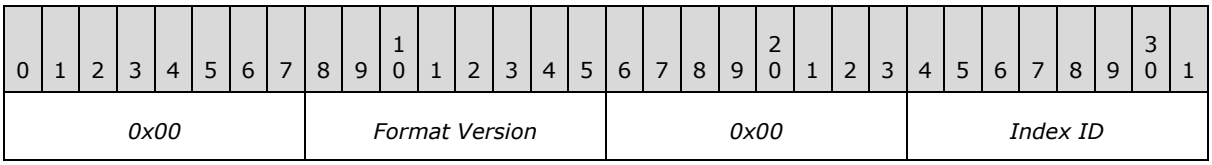


Number of Strings (4 bytes): A 32-bit unsigned integer representing the number of strings in the list. It MUST be located at the beginning of the file.

String Records (variable): A variable-length array of string records as specified in section 2.2.2.1.1. 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 Versioned Index Identifier

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



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.

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.

3 Protocol Details

There are three roles of this protocol: back-end database server, sender, and receiver.

Most of the messages sent and received in this protocol are stored procedure calls and the result sets they return. These sproc and result set messages are specified in section 3.1. There is one non-sproc-related message, a file transfer from sender to receiver, which is specified in section 2.1.

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 the other endpoints of the protocol.

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 (section 3.1.1.1) and a list of running tasks (section 3.1.1.2).

3.1.1.1 List of Ready Query Components

A list of zero or more query components which includes all of the query components in the Query Component Set ([MS-SRCHTP] section 3.1.1.3) whose State value ([MS-SRCHTP] section 3.1.1.3) is equal to any of the following:

1. Ready,
2. SplittingIndexes,
3. Reverting, or
4. IndexSplitDone,

as specified in [MS-SRCHTP] section 2.2.1.3, and are members of the **query topology** whose State ([MS-SRCHTP] section 3.1.1.3) value is Active ([MS-SRCHTP] section 2.2.1.2). The list does not contain any other query components.

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. A running task has the following properties:

taskType: The task type (section 2.2.1.2.1) of the propagation task.

senderID: The CrawlComponentNumber ([MS-SRCHTP] section 3.1.1.3) of the **crawl component** which created the propagation task.

catalogID: The catalog ID (section 2.2.1.2.2) of the full-text index catalog to which the propagation task applies.

list of completions: A list of zero or more query components which have completed the propagation task.

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

componentID: The versioned index identifier (section 2.2.3) of the full-text index component being added.

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

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

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.

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 exists within the back-end database server in a valid state.

3.1.4 Higher-Layer Triggered Events

None.

3.1.5 Message Processing Events and Sequencing Rules

The back-end database server does not initiate any communication. It only issues 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 completed propagation task. The **T-SQL** syntax for the stored procedure is as follows:

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

@SenderID: The sender ID (section 3.2.1.2) of the sender which created the propagation task.

@CatalogID: The catalog ID (section 2.2.1.2.2) representing the full-text index catalog to which the propagation task applies.

@TaskType: The task type (section 2.2.1.2.1) of the propagation task.

@ObjectID: If *@TaskType* is ComponentAddition (section 2.2.1.2.1), then *@ObjectID* MUST be either 0 or the versioned index identifier (section 2.2.3) of the full-text index component that was propagated. If *@TaskType* is StaticRankComputation (section 2.2.1.2.1), then *@ObjectID* MUST be 0.

When the back-end database server receives this message:

If the State ([MS-SRCHTP] section 3.1.1.3) of the crawl component which 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.

Return Code Values:

Value	Description
0	The task was added to the list of running tasks.
1	The task was not added to the list of running tasks because the crawl component was disabled.

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 completed by all query components in the list of ready query components (section 3.1.1.1). The T-SQL syntax for the stored procedure is as follows:

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

@SenderID: The sender ID (section 3.2.1.2) of the sender which created the propagation task.

@CatalogID: The catalog ID (section 2.2.1.2.2) of the full-text index catalog for which the caller will receive completed propagation tasks.

Return Code Values: An integer which MUST be 0.

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

1. If the State value ([MS-SRCHTP] section 3.1.1.3) of the crawl component which 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.

2. Otherwise the returned result set MUST contain exactly one result for each propagation task which has been completed by all query components in the list of ready query components (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:

```
SenderID            int,
CatalogID           int,
TaskType            int,
ObjectID            int,
{MaxWorkID}         int,
{BirthDate}         int
```

SenderID: The senderID (section 3.1.1.2) of the propagation task.

CatalogID: The catalogID (section 3.1.1.2) of the propagation task.

TaskType: The taskType (section 3.1.1.2) of the propagation task.

ObjectID: If taskType (section 3.1.1.2) is ComponentAddition (section 2.2.1.2.1), ObjectID MUST be the componentID (section 3.1.1.2) of the running propagation task. If taskType (section 3.1.1.2) is StaticRankComputation (section 2.2.1.2.1), ObjectID MUST be 0.

{MaxWorkID}: MUST be 0.

{BirthDate}: MUST be 0.

3.1.5.3 proc_MSS_PropagationIndexerGetReadyQueryComponents

The **proc_MSS_PropagationIndexerGetReadyQueryComponents** stored procedure is called to retrieve information about all query components in the list of ready query components (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 which MUST be 0.

Result Sets: MUST return exactly one Ready Query Components Result Set (section 3.1.5.3.1). This result set MUST contain exactly one result for each query component in the list of ready query components (section 3.1.1.2), 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 ([MS-SRCHTP] section 3.1.1.2) of the query component.

QueryComponentNumber: The QueryComponentNumber ([MS-SRCHTP] section 3.1.1.2) of the query component.

PartitionID: The PartitionID ([MS-SRCHTP] section 3.1.1.2) of the query component.

ShareName: The ShareName ([MS-SRCHTP] section 3.1.1.2) of the query component.

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 ID (section 3.2.1.2) of the calling sender.

@CatalogID: The catalog ID (section 2.2.1.2.2) of the full-text index catalog for which the caller will receive propagation tasks.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST return a Propagation Tasks Result Set as specified in section 3.1.5.6.1. The returned result set MUST contain exactly one result for each propagation task in the list of running tasks (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 (section 3.1.1.2) of the propagation task.

CatalogID: The catalogID (section 3.1.1.2) of the propagation task.

TaskType: The taskType (section 3.1.1.2) of the propagation task.

ObjectID: If taskType (section 3.1.1.2) is ComponentAddition (section 2.2.1.2.1), ObjectID MUST be the componentID (section 3.1.1.2) of the running propagation task. If taskType (section 3.1.1.2) is StaticRankComputation (section 2.2.1.2.1), ObjectID MUST be 0.

{MaxWorkID}: MUST be 0.

{BirthDate}: MUST be 0.

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 (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 ID (section 3.2.1.2) of the calling sender.

@CatalogID: The catalog ID (section 2.2.1.2.2) of the full-text index catalog to which the propagation task applies.

@TaskType: The task type (section 2.2.1.2.1) of the propagation task.

@ObjectID: If *@TaskType* is ComponentAddition (section 2.2.1.2.1), then *@ObjectID* MUST be either 0 or the versioned index identifier (section 2.2.3) of the full-text index component that is being propagated. If *@TaskType* is StaticRankComputation (section 2.2.1.2.1), then *@ObjectID* MUST be 0.

@MaxWorkID: If *@TaskType* is ComponentAddition (section 2.2.1.2.1, then *@MaxWorkID* MUST be either 0 or the maximum document identifier in the full-text index component being propagated. If *@TaskType* is not ComponentAddition (section 2.2.1.2.1), then *@MaxWorkID* MUST be 0.

@BirthDate: If *@TaskType* is ComponentAddition (section 2.2.1.2.1), then *@BirthDate* MUST be either 0 or the component birth date of the full-text index component being propagated. If *@TaskType* is not ComponentAddition (section 2.2.1.2.1, then *@BirthDate* MUST be 0.

When the back-end database server receives this message:

1. If the State value ([MS-SRCHTP] section 3.1.1.3) of the crawl component which created the propagation task is either Disabled or DisableForRemove, as specified in [MS-SRCHTP] section 2.2.1.7, then the back-end database server MUST return 1.

2. Otherwise,

    1. If the *@CatalogID*, *@TaskType*, and *@ObjectID* parameters match the catalogID, taskType, and objectID values of a propagation task in the list of running tasks (section 3.1.1.2), then the back-end database server MUST return 1.

    2. Otherwise the back-end database server MUST add a new propagation task to the list of running tasks, where catalogID (section 3.1.1.2) equals *@CatalogID*, taskType (section 3.1.1.2) equals *@TaskType*, objectID (section 3.1.1.2) equals *@ObjectID*, and startTime (section 3.1.1.2) equals the current local time in **datetime** format.

Return Code Values:

Value	Description
0	The propagation task was added.
1	No propagation task was added because the crawl component was disabled or a duplicate propagation task already existed.

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 which have not yet been completed by a particular query component. The T-SQL syntax for the stored procedure is as follows:

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

@CatalogID: The catalog ID (section 2.2.1.2.2) representing the full-text index catalog to which the retrieved propagation tasks will apply.

@ReceiverID: The receiver ID (section 3.3.1.1) of the calling receiver.

When the back-end database server receives this message:

If no query component with *@ReceiverID* is in the list of ready query components (section 3.1.1.1), then it MUST return 1.

Otherwise it MUST return 0.

Return Code Values:

Value	Description
0	A result set with the incomplete propagation tasks was returned.
1	No result set was returned because the query component was not in the list of ready query components.

Result Sets:

1. If no query component with *@ReceiverID* is in the list of ready query components, the back-end database server MUST NOT return any result sets.
2. Otherwise it MUST return exactly one Propagation Tasks Result Set (section 3.1.5.6.1). The result set MUST include exactly one result for each incomplete propagation task for the full-text index catalog identified by *@CatalogID*. A propagation task is incomplete if it has not been reported complete by at least one of the query components in the list of ready query components (section 3.1.1.1). The result set MUST NOT include any other results. The results MUST be ordered in ascending order, primarily by senderID values (section 3.1.5.6.1) and secondarily by birthDate values (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:

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

SenderID: The senderID (section 3.1.1.2) of the propagation task.

CatalogID: The catalogID (section 3.1.1.2) of the propagation task. This MUST be the same value as the input parameter *@CatalogID*.

TaskType: The taskType (section 3.1.1.2) of the propagation task.

ObjectID: For all results where the TaskType value is ComponentAddition (section 2.2.1.2.1), ObjectID MUST be the componentID (section 3.1.1.2) of the propagation task. For all results where the TaskType value is not ComponentAddition (section 2.2.1.2.1), ObjectID MUST be 0.

MaxWorkID: For all results where the TaskType value is ComponentAddition (section 2.2.1.2.1), MaxWorkID MUST be the maxDocID value (section 3.1.1.2) of the propagation task. For all results where the TaskType value is not ComponentAddition (section 2.2.1.2.1), MaxWorkID MUST be 0.

BirthDate: For all results where the TaskType value is ComponentAddition (section 2.2.1.2.1), BirthDate MUST be the birthDate value (section 3.1.1.2) of the propagation task. For all results where the TaskType value is not ComponentAddition (section 2.2.1.2.1), BirthDate MUST be 0.

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 (section 3.1.1.2) that were created by the calling sender. The T-SQL syntax for the stored procedure is as follows:

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

@SenderID: The sender ID (section 3.2.1.2) of the calling sender.

@CatalogID: The catalog ID (section 2.2.1.2.2) of the full-text index catalog for which the propagation tasks will be removed from the list of running tasks (section 3.1.1.2).

Return Code Values: An integer which MUST be 0.

Result Sets: MUST NOT return any result set.

3.1.5.8 proc_MSS_PropagationIndexerGetActiveIndexPartitionGuids

The **proc_MSS_PropagationIndexerGetActiveIndexPartitionGuids** stored procedure is called to get information about the **index partitions** in the active query topology. The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_PropagationIndexerGetActiveIndexPartitionGuids;
```

Return Code Values: An integer which MUST be 0.

Result Sets: MUST return a Partitions Result Set as specified in section 3.1.5.8.1. It MUST contain a result for each index partition ([MS-SRCHTP] section 3.1.1.2) in the query topology in the query topology set ([MS-SRCHTP] section 3.1.1.2) whose State value ([MS-SRCHTP] section 3.1.1.2) is Active ([MS-SRCHTP] section 2.2.1.2).

3.1.5.8.1 Partitions Result Set

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

```
Ordinal             tinyint,
PartitionID         uniqueidentifier
```

Ordinal: The ordinal value ([MS-SRCHTP] section 3.1.1.2) of the index partition.

PartitionID: The identifier value ([MS-SRCHTP] section 3.1.1.2) of the index partition.

3.1.5.9 proc_MSS_PropagationIndexerGetActiveIndexPartitionHashes

The **proc_MSS_PropagationIndexerGetActiveIndexPartitionHashes** stored procedure is called to get information about the query topology in the active query topology. The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_PropagationIndexerGetActiveIndexPartitionHashes;
```

Return Code Values: An integer which MUST be 0.

Result Sets: MUST return a Partition Hashes Result Set as specified in section 3.1.5.9.1. It MUST contain a result for each item in the index partition hash set ([MS-SRCHTP] section 3.1.1.2 of the query topology in the query topology set ([MS-SRCHTP] section 3.1.1.2) whose State value ([MS-SRCHTP] section 3.1.1.2) is Active ([MS-SRCHTP] section 2.2.1.2). It MUST NOT contain any other results.

3.1.5.9.1 Partition Hashes

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

```
Hash                tinyint,
Ordinal             tinyint
```

Hash: A value between 0 and 255.

Ordinal: The Ordinal value ([MS-SRCHTP] section 3.1.1.2) of the index partition to which the value of Hash is associated in the Index Partition Hash Set ([MS-SRCHTP] section 3.1.1.2).

3.1.5.10 proc_MSS_PropagationQueryComponentReportTaskReady

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

```
PROCEDURE proc_MSS_PropagationQueryServerReportTaskReady(
   @SenderID           int,
   @CatalogID          int,
   @ReceiverID         int,
   @TaskType           int,
   @ObjectID           int
);
```

@SenderID: The sender ID (section 3.2.1.2) of the sender which created the propagation task.

@CatalogID: The catalog ID (section 2.2.1.2.2) of the full-text index catalog to which the propagation task applies.

@ReceiverID: The receiver ID (section 3.3.1.1) of the calling receiver.

@TaskType: Any value of task type (section 2.2.1.2.1).

@ObjectID: If *@TaskType* is ComponentAddition (section 2.2.1.2.1), then *@ObjectID* MUST be the versioned index identifier (section 2.2.3) of the full-text index component that is being propagated. If *@TaskType* is StaticRankComputation (section 2.2.1.2.1), then *@ObjectID* MUST be 0.

When the back-end database server receives this message,

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

2. Otherwise,

    1. If the query component with QueryComponentNumber ([MS-SRCHTP] section 3.1.1.2) equal to *@ReceiverID* 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.

    2. Otherwise, the back-end database server MUST add the query component with QueryComponentNumber ([MS-SRCHTP] section 3.1.1.2) 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*. The list of completions, list of running tasks, catalogID, taskType, and objectID are specified in section 3.1.1.

Return Code Values:

Value	Description
0	Successful execution.
1	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.

Result Set: MUST NOT return any result set.

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 ID

An integer which 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 that will receive full-text index component messages (section 2.2.2). Each ready query component has the following properties:

serverName: The ServerName of the query component as specified in [MS-SRCHTP] section 3.1.1.2.

shareName: The ShareName of the query component as specified in [MS-SRCHTP] section 3.1.1.2.

3.2.1.4 List of Completed Tasks

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

catalogID: The catalog ID (section 2.2.1.2.2 of the full-text index catalog to which the propagation task applies.

taskType: The task type of the propagation task (section 2.2.1.2.1).

objectID: If **taskType** is ComponentAddition, the versioned index identifier (section 2.2.3) of the full-text index component being added on the query components. For all other values of **taskType**, the value is not used.

The sender uses this list to record all propagation tasks for which it MUST send proc_MSS_PropagationIndexerCleanUpTablesForTask messages.

3.2.2 Timers

None.

3.2.3 Initialization

None.

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 (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 (section 3.2.1.3) to match the back-end database server's list of ready query components (section 3.1.1.1).

3.2.5.2 Receiving a Ready Query Components Result Set

This result set (section 3.1.5.3.1) is received automatically after calling the **proc_MSS_PropagationIndexerGetReadyQueryComponents** stored procedure (section 3.1.5.2). The sender MUST replace its current list of ready query components (section 3.2.1.3) with exactly one ready query component for each received result, where:

1. serverName (section 3.2.1.3) is set to the ServerName value (section 3.1.5.3.1) of the result,
2. componentName (section 3.2.1.3) is set to the QueryComponentName value (section 3.1.5.3.1) of the result, and
3. shareName (section 3.2.1.3) is set to the ShareName value (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 (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.

When a full-text index component is generated for a full-text index catalog on the sender, the sender MUST perform the following actions:

1. The sender generates a full-text index component, exactly 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 ID (section 3.2.1.2) plus a period, like "000A.", and an additional suffix ".cp".

2. The sender generates a propagation list file (section 2.2.2.1) containing the file names (not the full file system paths) of each of the files contained in the propagated full-text index component message other than the propagation list file itself.

3. The sender copies the duplicated full-text index component files and the propagation list file (section 2.2.2.1) to a path relative to serverName (section 3.2.1.3) and shareName (section 3.2.1.3) of each ready query component in the list of ready query components (section 3.2.1.3). The destination path should be \\<server>\<share>\<application>-query-<receiverID>\Projects\<catalog>\Indexer\CiFiles\<file>, where

4. <server> is the serverName (section 3.2.1.3) of the query component,

5. <share> is the shareName (section 3.2.1.3) of the query component,

6. <application> is the search application name (section 4.1.1.2.1),

7. <receiverID> is the QueryComponentNumber ([MS-SRCHTP] section 3.1.1.2) of the query component,

8. <catalog> is

    1. "Portal_Content" if the full-text index catalog is the main catalog as specified in [MS-CIFO] section 2.18.1, or

    2. "AnchorProject" if the full-text index catalog is the anchor text catalog as specified in [MS-CIFO] section 2.18.2, and

9. <file> is the file name specified above.

10. It MUST then call the **proc_MSS_PropagationIndexerInsertNewTask** stored procedure as specified in section 3.1.5.5.

3.2.5.4 Sending a proc_MSS_PropagationIndexerInsertNewTask Message

This is the first message of the propagation sequence for cleaning (specified below) 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 (section 3.1.5.5):

1. Component addition. The sender calls the **proc_MSS_PropagationIndexerInsertNewTask** stored procedure with the following parameters:

    1. *@SenderID* MUST be the sender ID (section 3.2.1.2) of this sender.

    2. *@CatalogID* MUST be the catalog ID (section 2.2.1.2.2) for the full-text index component.

    3. *@TaskType* MUST be ComponentAddition (section 2.2.1.2.1).

    4. *@ObjectID* MUST be the versioned index identifier (section 2.2.3) of the full-text index component.

    5. *@MaxWorkID* MUST be the maximum document identifier in the full-text index component.

6. *@BirthDate* MUST be the component birth date of the **document set** of the full-text index component.

2. Cleaning. It is often desirable to ensure that all query components have completed all tasks before inserting another one. For this, the sender SHOULD call the **proc_MSS_PropagationIndexerInsertNewTask** stored procedure with the following parameters:

   1. *@SenderID* MUST be the sender ID (section 3.2.1.2) of this sender.

   2. *@CatalogID* MUST be the catalog ID (section 2.2.1.2.2) for the full-text index component.

   3. *@TaskType* MUST be ComponentAddition (section 2.2.1.2.1).

   4. *@ObjectID* MUST be 0.

   5. *@MaxWorkID* MUST be 0.

   6. *@BirthDate* MUST be 0.

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

3. Static rank computation. The sender calls the **proc_MSS_PropagationIndexerInsertNewTask** stored procedure with the following parameters:

   1. *@SenderID* MUST be the sender ID (section 3.2.1.2) of this sender.

   2. *@CatalogID* MUST be the catalog ID (section 2.2.1.2.2) of the full-text index catalog over which static rank computation will be performed.

   3. *@TaskType* MUST be StaticRankComputation (section 2.2.1.2.1).

   4. *@ObjectID* MUST be 0.

   5. *@MaxWorkID* MUST be 0.

   6. *@BirthDate* MUST be 0.

3.2.5.5 Sending a proc_MSS_PropagationIndexerGetCompletedTasks Message

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

The sender MUST call the **proc_MSS_PropagationIndexerGetCompletedTasks** stored procedure (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 will not prevent the successful execution of propagation tasks.

1. *@SenderID* (section 3.1.5.2) MUST be the sender ID (section 3.2.1.2) of this sender.

2. *@CatalogID* (section 3.1.5.2) MUST be the identifier of the full-text index catalog.

3.2.5.6 Receiving a Completed Tasks Result Set Message

A Completed Tasks Result Set (section 3.1.5.2.1) is received automatically following any call to the **proc_MSS_PropagationIndexerGetCompletedTasks** stored procedure (section 3.1.5.2). The

full-text index catalog to which the result set applies is evident in the CatalogID value (section 3.1.5.2.1) of each result in the result set. For this full-text index catalog, the sender MUST replace its list of completed tasks (section 3.2.1.4) with a new list containing one completed task for each result in this result set, where

1. catalogID (section 3.2.1.4) is set to the CatalogID value (section 3.1.5.2.1) of the result,
2. taskType (section 3.2.1.4) is set to the TaskType value (section 3.1.5.2.1) of the result, and
3. objectID (section 3.2.1.4) is set to the ObjectID value (section 3.1.5.2.1) of the result.

3.2.5.7 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 (section 3.2.1.4), the **proc_MSS_PropagationIndexerCleanUpTablesForTask** stored procedure (section 3.1.5.1) MUST be called once for each completed task, using the following parameters:

1. *@CatalogID* (section 3.1.5.1) is set to the catalogID value (section 3.2.1.4) of the completed task.
2. *@TaskType* (section 3.1.5.1) is set to the taskType value (section 3.2.1.4) of the completed task.
3. *@ObjectID* (section 3.1.5.1) is set to the objectID value (section 3.2.1.4) of the completed task.

3.2.5.8 Sending a proc_MSS_PropagationIndexerGetTasks Message

Not part of any sequence.

The sender calls the **proc_MSS_PropagationIndexerGetTasks** stored procedure (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 (section 3.1.1.2). The *@SenderID* parameter (section 3.1.5.4) MUST be the sender's Sender ID (section 3.2.1.2). The *@CatalogID* parameter (section 3.1.5.4) MUST be the catalog ID (section 2.2.1.2.2) of a full-text index catalog.

3.2.5.9 Receiving a Propagation Tasks Result Set

This result set is received automatically after calling the **proc_MSS_PropagationIndexerGetTasks** stored procedure (section 3.2.5.8).

Receiving this message MUST NOT affect the state of the sender that is specified in section 3.2.1.

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. 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 ID

An integer which uniquely identifies the receiver. This MUST be equal to the QueryComponentNumber of the query component, 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:

catalogID: The catalog ID (section 2.2.1.2.2) of the full-text index catalog to which the propagation task applies.

taskType: The task type of the propagation task (section 2.2.1.2.1).

objectID: If taskType is ComponentAddition (section 2.2.1.2.1), the versioned index identifier (section 2.2.3) of the full-text index component being added on the query components. For all other values of **taskType**, the value is not used.

maxWorkID: If taskType is ComponentAddition (section 2.2.1.2.1), the maximum document identifier in the full-text index component. For all other values of taskType, the value is not used.

birthDate: If taskType is ComponentAddition (section 2.2.1.2.1), 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 State

None.

3.3.2 Timers

None.

3.3.3 Initialization

None.

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 for a propagation sequence begins with this call.

The receiver MUST call the **proc_MSS_PropagationQueryComponentPickUpNewPropagationItems** stored procedure (section 3.1.5.10) 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 will not prevent the successful execution of propagation tasks. The procedure MUST be called with the following parameters:

- *@ReceiverID* (section 3.1.5.10) MUST be the receiver ID (section 3.3.1.1) of this receiver.
- *@CatalogID* (section 3.1.5.10) MUST be the identifier of the full-text index catalog.

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 (section 3.1.5.10). The full-text index catalog to which the result set applies is specified in the CatalogID value (section 3.1.5.4.1) of each result in the result set. For this full-text index catalog, the **query component** MUST replace its list of incomplete tasks (section 3.3.1.2) with a new list that contains one incomplete task for each result in this result set, where:

1. catalogID (section 3.3.1.2) is set to the CatalogID value (section 3.1.5.4.1) of the result,
2. taskType (section 3.3.1.2) is set to the TaskType value (section 3.1.5.4.1) of the result,
3. objectID (section 3.3.1.2) is set to the ObjectID value (section 3.1.5.4.1) of the result,
4. maxWorkID (section 3.3.1.2) is set to the MaxWorkID (section 3.1.5.4.1) value of the result, and
5. birthDate (section 3.3.1.2) is the BirthDate value (section 3.1.5.4.1) of the result.

3.3.5.3 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 (section 3.1.5.10) MUST be called once for each incomplete task in the receiver's list of incomplete tasks (section 3.3.1.2), where:

1. *@ReceiverID* (section 3.1.5.10) is set to receiver ID (section 3.3.1.1),
2. *@CatalogID* (section 3.1.5.10) is set to the catalogID value (section 3.3.1.1) of the incomplete task,
3. *@TaskType* (section 3.1.5.10) is set to the taskType value (section 3.3.1.1) of the incomplete task, and
4. *@ObjectID* (section 3.1.5.10) is set to the objectID value (section 3.3.1.1) of the incomplete task.

3.3.6 Timer Events

None.

3.3.7 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 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 two query components:

1. REC-1
    1. QueryComponentNumber is 0.
    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 ID

The ID is 0.

4.1.1.2.3 List of Ready Query Components

The list contains two query components:

1. REC-1
    1. receiverID is 0.
    2. serverName is REC-1.
    3. ShareName is "4c436ee0-b809-4e8a-b00b-be776306e0ee-query-0".
2. REC-2
    1. receiverID is 1.
    2. serverName is REC-2.
    3. 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 ID

The ID is 0.

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 ID

The ID is 1.

4.1.1.4.2 List of Incomplete Tasks

The list is empty.

4.1.2 Sequence

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.

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 0.
2. DB-1 replies with an empty completed tasks result set, indicating that there are no completed tasks for the main catalog.

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 0.

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

5. REC-2 polls DB-1 every 30 seconds by calling the **proc_MSS_PropagationQueryComponentPickUpNewPropagationItems** stored procedure with *@CatalogID* set to 1. DB-1 returns 0.

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 will propagate the component. The full-text index component has index ID 0x0001001A, versioned index ID 0x0054001A, maximum document identifier 471952, and component birth date 414.

- SEN-1 writes the full-text index component files listed in the table below. All file names begin with the sender ID 0, contain one of the file names of a full-text index component, and end with the "cp" extension.

File name
0000.0001001A.ci.cp
0000.0001001A.dir.cp
0000.0001001A.bsi.cp
0000.0001001A.bsd.cp
0000.0001001A.csi.cp
0000.0001001A.csd.cp
0000.0001001A.wid.cp
0000.0001001A.list.cp

to both of the following file paths:

File share
\\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

1. SEN-1 calls the **proc_MSS_PropagationIndexerInsertNewTask** stored procedure with the following parameters:

    1. *@SenderID* is set to 0

    2. *@CatalogID* is set to 1

    3. *@TaskType* is set to ComponentAddition

4. *@ObjectID* is set to 5505050 (hexadecimal equivalent: 0x0054001A)
5. *@MaxWorkID* is set to 471952
6. *@BirthDate* is set to 414

   DB-1 returns 0.

2. REC-1 calls the **proc_MSS_PropagationQueryComponentPickUpNewPropagationItems** stored procedure with *@ReceiverID* set to 0 and *@CatalogID* set to 1. DB-1 returns 0.
3. DB-1 sends a propagation tasks result set with one result where:
   1. SenderID is set to 0
   2. CatalogID is set to 1
   3. TaskType is set to ComponentAddition
   4. ObjectID is set to 5505050 (hexadecimal equivalent: 0x0054001A)
   5. MaxWorkID is set to 471952
   6. Birthdate is set to 414
4. REC-1 applies the full-text index component and calls the **proc_MSS_PropagationQueryComponentReportTaskReady** stored procedure with the following parameters:
   1. *@ReceiverID* is set to 0
   2. *@SenderID* is set to 0
   3. *@CatalogID* is set to 1
   4. *@TaskType* is set to ComponentAddition
   5. *@ObjectID* is set to 5505050

DB-1 returns 0.

1. SEN-1 polls DB-1 again by calling the **proc_MSS_PropagationIndexerGetCompletedTasks** stored procedure with *@CatalogID* set to 1. DB-1 returns 0.
2. DB-1 sends an empty completed tasks result set, indicating that there are no completed tasks for the main catalog.
3. REC-2 calls the **proc_MSS_PropagationQueryComponentPickUpNewPropagationItems** stored procedure with *@ReceiverID* set to 1 and *@CatalogID* set to 1.
4. DB-1 returns the following propagation tasks result set:
   1. SenderID is set to 0
   2. CatalogID is set to 1
   3. TaskType is set to ComponentAddition
   4. ObjectID is set to 5505050

5. MaxWorkID is set to 471952
6. Birthdate is set to 414

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

   DB-1 returns 0.
6. The index server SEN-1 polls DB-1 again by calling the **proc_MSS_PropagationIndexerGetCompletedTasks** stored procedure with *@SenderID* set to 0 and *@CatalogID* set to 1. DB-1 returns 0.
7. DB-1 returns the following completed tasks result set:
   1. SenderID is set to 0
   2. CatalogID is set to 1
   3. TaskType is set to ComponentAddition
   4. ObjectID is set to 5505050
   5. {MaxWorkID} is set to 0
   6. {Birthdate} is set to 0.
8. The index server SEN-1 then calls the **proc_MSS_PropagationIndexerCleanUpTablesForTask** stored procedure with the following parameters:
   1. *@SenderID* is set to 0
   2. *@CatalogID* is set to 1
   3. *@TaskType* is set to ComponentAddition
   4. *@ObjectID* is set to 5505050.

   DB-1 returns 0.
9. 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 ([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.

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® Office SharePoint® Server 2007
- Microsoft® SQL Server® 2005
- Microsoft® SQL Server® 2008
- Microsoft® SQL Server® 2008 R2
- Windows® SharePoint® Services 3.0

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

N

O

P

R

S

T

V