[MS-FSQRC]:
Query and Result Configuration 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
11/06/2009	0.1	Major	Initial Availability
02/19/2010	1.0	Major	Updated and revised the technical content
03/31/2010	1.01	Editorial	Revised and edited the technical content
04/30/2010	1.02	Editorial	Revised and edited the technical content
06/07/2010	1.03	Editorial	Revised and edited the technical content
06/29/2010	1.04	Editorial	Changed language and formatting in the technical content.
07/23/2010	1.04	No change	No changes to the meaning, language, or formatting of the technical content.
09/27/2010	1.04	No change	No changes to the meaning, language, or formatting of the technical content.
11/15/2010	1.04	No change	No changes to the meaning, language, or formatting of the technical content.
12/17/2010	1.04	No change	No changes to the meaning, language, or formatting of the technical content.

Table of Contents

1 Introduction

This document specifies the Query and Result Configuration Protocol, which loads updated configuration information for subsystems of a **query processing** component.

1.1 Glossary

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

abstract object reference (AOR)
administration component
automaton
base port
client proxy
dictionary
FAST Search Interface Definition Language (FSIDL)
name server
property extraction
query processing
query refinement
token

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-FSLRDS] Microsoft Corporation, "Linguistic Resource Data Structure", November 2009.

[MS-FSMW] Microsoft Corporation, "Middleware Protocol Specification", November 2009.

[MS-FSRS] Microsoft Corporation, "Resource Store Protocol Specification", November 2009.

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

1.2.2 Informative References

[MS-OFCGLOS] Microsoft Corporation, "Microsoft Office Master Glossary", June 2008.

1.3 Overview

This protocol is used by the **administration component** to request that the query processing component retrieve updated information from a resource store.

The protocol is transaction-based, and each message contains a transaction identifier. The protocol client activates a transaction by calling **commit**, and cancels a transaction by calling **abort**.

The protocol client requests that the query processing component retrieve updated spell checking **automata** and **dictionaries** by calling **prepare_set_query_pipeline**. To request that the query processing component retrieve updated **property extraction** exceptions, the protocol server calls **prepare_set_result_pipeline**.

1.4 Relationship to Other Protocols

The interfaces are described using the **FAST Search Interface Definition Language (FSIDL)**, as described in [MS-FSMW]. The messages are transported using the HTTP-based interoperability framework described in [MS-FSMW].

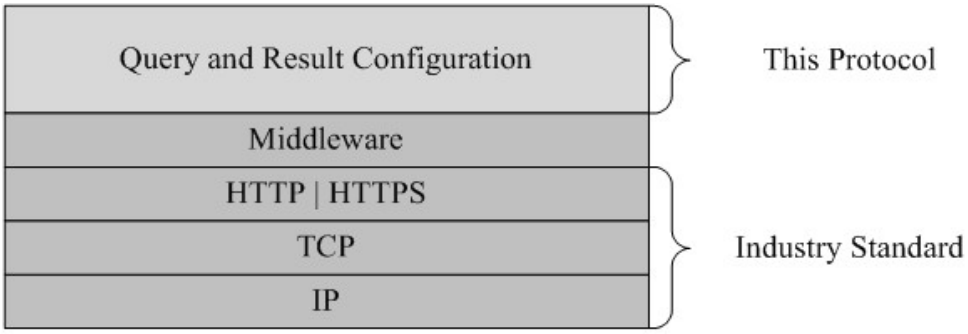

Figure 1: This protocol in relation to other protocols

1.5 Prerequisites/Preconditions

The protocol client and protocol server have the location and connection information of the shared **name server**.

1.6 Applicability Statement

These protocols are applicable for search applications that remotely restart query processing component subsystems.

1.7 Versioning and Capability Negotiation

This protocol is connectionless. The interface version is specified in every message that uses this protocol. For more information about version numbers, see section 3.1.3.

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2 Messages

2.1 Transport

The messages supported by the interfaces in the following subsections MUST be sent as HTTP POST messages, as specified in [MS-FSMW].

2.2 Common Data Types

None.

3 Protocol Details

The client side of this protocol is simply a pass-through. That is, no additional timers or other states are required on the client side of this protocol. Calls made by the higher-layer protocol or application are passed directly to the transport, and the results returned by the transport are passed directly back to the higher-layer protocol or application.

3.1 Server Details

A protocol will retrieve the pertinent configuration files whenever the protocol server receives either a **prepare_set_query_pipeline** or **prepare_set_result_pipeline** message. If the protocol server receives a **commit** message, it will activate any updates that are associated with the specified transaction identifier. An **abort** message will cause the protocol server to discard any configuration updates that are associated with the specified transaction identifier.

3.1.1 Abstract Data Model

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

active transactions: A list of active transactions.

configuration changes: A set of configuration changes for each active transaction. This contains the set of configuration changes to apply if the transaction is committed.

3.1.2 Timers

None.

3.1.3 Initialization

The protocol server MUST use the Middleware **bind** method to register **searchservice::configuration** and **core::transaction_client** server objects in the name server, as specified in [MS-FSMW] section 3.4.4.2.

The parameters for the **bind** method are encapsulated in an **abstract object reference (AOR)**, as specified in [MS-FSMW] section 2.2.18.

host: This string MUST contain the host name of the server object on the protocol server. The value is implementation specific, and is specified in the configuration file %FASTSEARCH%\etc\Node.xml.

port: This MUST be an integer value that contains the port number of the server object on the protocol server. The value MUST be **base port** + 390.

interface_type: A string that MUST contain either "searchservice::configuration" or "core::transaction_client".

interface_version: A string that MUST contain "5.1".

object_id: An integer that MUST be unique for each server object**.**

name: A string that MUST contain "fds/searchservice/*hostname_port*", where *hostname* MUST be the fully qualified host name of the system on which the protocol server runs, and *port* MUST be base port + 280.

3.1.4 Message Processing Events and Sequencing Rules

The message type is determined at the middleware level. The middleware MUST call the correct method of a server object that implements an interface.

Some of the following methods use generic middleware exceptions to send error messages to the protocol client. The exceptions can be thrown from any method, as specified in [MS-FSMW], and consequently, are not defined in the FSIDL method signatures.

Method	Description
searchservice::configuration::prepare_set_query_pipeline	Causes the query processing component to retrieve updated information to use during query processing.
searchservice::configuration::prepare_set_result_pipeline	Causes the query processing component to retrieve updated information to use in result post-processing.
core::transaction_client::commit	Commits the specified transaction, and activates any changes loaded by **prepare_set_query_pipeline** or **prepare_set_result_pipeline**.
core::transaction_client::abort	Cancels the specified transaction.

3.1.4.1 searchservice::configuration::prepare_set_query_pipeline

The **prepare_set_query_pipeline** method causes the protocol server to retrieve updated information for use during query processing. Updates are retrieved from a resource store, as specified in [MS-FSRS].

If the specified transaction identifier is not already active, it MUST be added to **active transactions**, as specified in section 3.1.1. Configuration updates MUST be retrieved and added to **configuration changes**, as specified in section 3.1.1.

This method MUST update the following spell checking automaton files:

- dictionaries/spellcheck/ar_spell_iso8859_6.xml.gz
- dictionaries/spellcheck/bg_spell_iso8859_5.xml.gz
- dictionaries/spellcheck/ca_spell_iso8859_1.xml.gz
- dictionaries/spellcheck/cs_spell_iso8859_2.xml.gz
- dictionaries/spellcheck/da_spell_iso8859_1.xml.gz
- dictionaries/spellcheck/de_spell_iso8859_1.xml.gz
- dictionaries/spellcheck/el_spell_iso8859_7.xml.gz
- dictionaries/spellcheck/en_spell_iso8859_1.xml.gz

- dictionaries/spellcheck/es_spell_iso8859_1.xml.gz
- dictionaries/spellcheck/et_spell_iso8859_13.xml.gz
- dictionaries/spellcheck/fi_spell_iso8859_1.xml.gz
- dictionaries/spellcheck/fr_spell_iso8859_1.xml.gz
- dictionaries/spellcheck/he_spell_iso8859_8.xml.gz
- dictionaries/spellcheck/hi_spell_fiscii.xml.gz
- dictionaries/spellcheck/hr_spell_iso8859_2.xml.gz
- dictionaries/spellcheck/hu_spell_iso8859_2.xml.gz
- dictionaries/spellcheck/id_spell_iso8859_1.xml.gz
- dictionaries/spellcheck/is_spell_iso8859_1.xml.gz
- dictionaries/spellcheck/it_spell_iso8859_1.xml.gz
- dictionaries/spellcheck/ko_spell_korean1b.xml.gz
- dictionaries/spellcheck/lt_spell_iso8859_13.xml.gz
- dictionaries/spellcheck/lv_spell_iso8859_13.xml.gz
- dictionaries/spellcheck/ms_spell_iso8859_1.xml.gz
- dictionaries/spellcheck/nb_spell_iso8859_1.xml.gz
- dictionaries/spellcheck/nl_spell_iso8859_1.xml.gz
- dictionaries/spellcheck/nn_spell_iso8859_1.xml.gz
- dictionaries/spellcheck/pl_spell_iso8859_2.xml.gz
- dictionaries/spellcheck/pt_spell_iso8859_1.xml.gz
- dictionaries/spellcheck/ro_spell_iso8859_2.xml.gz
- dictionaries/spellcheck/ru_spell_iso8859_5.xml.gz
- dictionaries/spellcheck/sk_spell_iso8859_2.xml.gz
- dictionaries/spellcheck/sl_spell_iso8859_2.xml.gz
- dictionaries/spellcheck/spellcheck_exclusion_any.xml
- dictionaries/spellcheck/sr_spell_iso8859_5.xml.gz
- dictionaries/spellcheck/sv_spell_iso8859_1.xml.gz
- dictionaries/spellcheck/tr_spell_iso8859_9.xml.gz
- dictionaries/spellcheck/uk_spell_iso8859_5.xml.gz
- dictionaries/spellcheck/ur_spell_windows_1256.xml.gz

This method MUST update the following spell checking exception automaton files:

- dictionaries/spellcheck/ar_exceptions_iso8859_6.xml.gz
- dictionaries/spellcheck/bg_exceptions_iso8859_5.xml.gz
- dictionaries/spellcheck/ca_exceptions_iso8859_1.xml.gz
- dictionaries/spellcheck/cs_exceptions_iso8859_2.xml.gz
- dictionaries/spellcheck/da_exceptions_iso8859_1.xml.gz
- dictionaries/spellcheck/de_exceptions_iso8859_1.xml.gz
- dictionaries/spellcheck/el_exceptions_iso8859_7.xml.gz
- dictionaries/spellcheck/en_exceptions_iso8859_1.xml.gz
- dictionaries/spellcheck/es_exceptions_iso8859_1.xml.gz
- dictionaries/spellcheck/et_exceptions_iso8859_13.xml.gz
- dictionaries/spellcheck/fi_exceptions_iso8859_1.xml.gz
- dictionaries/spellcheck/fr_exceptions_iso8859_1.xml.gz
- dictionaries/spellcheck/he_exceptions_iso8859_8.xml.gz
- dictionaries/spellcheck/hi_exceptions_fiscii.xml.gz
- dictionaries/spellcheck/hr_exceptions_iso8859_2.xml.gz
- dictionaries/spellcheck/hu_exceptions_iso8859_2.xml.gz
- dictionaries/spellcheck/id_exceptions_iso8859_1.xml.gz
- dictionaries/spellcheck/is_exceptions_iso8859_1.xml.gz
- dictionaries/spellcheck/it_exceptions_iso8859_1.xml.gz
- dictionaries/spellcheck/ko_exceptions_korean1b.xml.gz
- dictionaries/spellcheck/lt_exceptions_iso8859_13.xml.gz
- dictionaries/spellcheck/lv_exceptions_iso8859_13.xml.gz
- dictionaries/spellcheck/ms_exceptions_iso8859_1.xml.gz
- dictionaries/spellcheck/nb_exceptions_iso8859_1.xml.gz
- dictionaries/spellcheck/nl_exceptions_iso8859_1.xml.gz
- dictionaries/spellcheck/nn_exceptions_iso8859_1.xml.gz
- dictionaries/spellcheck/pl_exceptions_iso8859_2.xml.gz
- dictionaries/spellcheck/pt_exceptions_iso8859_1.xml.gz
- dictionaries/spellcheck/ro_exceptions_iso8859_2.xml.gz

- dictionaries/spellcheck/ru_exceptions_iso8859_5.xml.gz
- dictionaries/spellcheck/sk_exceptions_iso8859_2.xml.gz
- dictionaries/spellcheck/sl_exceptions_iso8859_2.xml.gz
- dictionaries/spellcheck/sr_exceptions_iso8859_5.xml.gz
- dictionaries/spellcheck/sv_exceptions_iso8859_1.xml.gz
- dictionaries/spellcheck/tr_exceptions_iso8859_9.xml.gz
- dictionaries/spellcheck/uk_exceptions_iso8859_5.xml.gz
- dictionaries/spellcheck/ur_exceptions_windows_1256.xml.gz

This method MUST update the following spell checking dictionary file:

- dictionaries/spellcheck/spellcheck_exclusion_any.xml

For more information about automaton files, see [MS-FSLRDS], section 2.1.

For more information about dictionary files, see [MS-FSLRDS], section 2.2.

The query processing component begins using the updated information when the protocol client calls **core::transaction_client::commit**, as specified in section 3.1.4.3.

The protocol server MUST raise the exception **core::unable_to_prepare_exception** if query processing cannot be configured with updated information.

The structure of this method, as specified in Appendix A, is as follows:

```
void prepare_set_query_pipeline(in long transaction_id)
  raises (core::unable_to_prepare_exception);
```

transaction_id: A number that identifies the transaction. All calls that are associated with the same transaction MUST use the same number.

3.1.4.2 searchservice::configuration::prepare_set_result_pipeline

The **prepare_set_result_pipeline** method causes the protocol server to retrieve updated information for use during result post-processing. Updates are retrieved from a resource store, as specified in [MS-FSRS].

If the specified transaction identifier is not already active, it MUST be added to **active transactions**, as specified in section 3.1.1. Configuration updates MUST be retrieved and added to **configuration changes**, as specified in section 3.1.1.

This method MUST update the following dictionary files that specify property extraction **tokens** to exclude from **query refinement**:

- dictionaries/matching/companies_exclusion_any.xml
- dictionaries/matching/locations_exclusion_any.xml
- dictionaries/matching/personnames_exclusion_any.xml

For more information about dictionary files, see [MS-FSLRDS], section 2.2.

The query processing component begins using the updated information when the protocol client calls **core::transaction_client::commit**, as specified in section 3.1.4.3.

The protocol server MUST raise the exception **core::unable_to_prepare_exception** if result processing cannot be configured with updated information.

The structure of this method, as specified in Appendix A, is as follows:

```
void prepare_set_result_pipeline(in long transaction_id)
  raises (core::unable_to_prepare_exception);
```

transaction_id: A number that identifies the transaction. All calls that are associated with the same transaction MUST use the same number.

3.1.4.3 core::transaction_client::commit

The **commit** method causes the query processing component to activate any changes loaded by the **prepare_set_query_pipeline** method or the **prepare_set_result_pipeline** method for the specified transaction, and disassociates the changes from that transaction.

The query processing component MUST activate the transaction's **configuration changes**, as specified in section 3.1.1. The specified transaction identifier MUST be removed from **active transactions**, as specified in section 3.1.1.

The structure of this method, as specified in Appendix A, is as follows:

```
void commit(in long transaction_id);
```

transaction_id: A number that identifies the transaction. All calls that are associated with the same transaction MUST use the same number.

3.1.4.4 core::transaction_client::abort

The **abort** method causes the query processing component to discard any changes loaded by the **prepare_set_query_pipeline** method or the **prepare_set_result_pipeline** method for the specified transaction, and disassociates the changes from that transaction.

The query processing component MUST discard the transaction's **configuration changes**, as specified in section 3.1.1. The specified transaction identifier MUST be removed from **active transactions**, as specified in section 3.1.1.

The structure of this method, as specified in Appendix A, is as follows:

```
void abort(in long transaction_id);
```

transaction_id: A number that identifies the transaction. All calls that are associated with the same transaction MUST use the same number.

3.1.5 Timer Events

None.

3.1.6 Other Local Events

None.

4 Protocol Examples

This example describes how to use the **prepare_set_query_pipeline** method.

Initially, the query processing component, acting as the protocol server in this transaction, creates server objects that implement the **configuration** and **transaction_client** interfaces. The protocol server then binds the server object to the name server. Subsequently, the protocol client acquires a **client proxy** for the **configuration** and **transaction_client** server objects. It does so by resolving the server object in the name server. This is possible because the location of the shared name server and the symbolic name of the server object were previously specified. That information is known to both the protocol client and the protocol server.

The protocol client then selects a transaction identifier – 0 in this example – which calls the **prepare_set_query_pipeline** method, and then calls the **commit** method.

4.1 Code

4.1.1 Protocol Server Initialization

```
SET server_object_instance TO INSTANCE OF configuration SERVER OBJECT
SET server_object_id TO UNIQUE INTEGER
SET server_object_host TO "myserver.mydomain.com"
SET server_object_port TO "1234"
SET server_object_interface_type TO "searchservice::configuration"
SET server_object_interface_version TO = 5.1
SET server_object_name TO "fds/searchservice/myserver.mydomain.com_13280"
SET server_object_aor TO server_object_host, server_object_port,
server_object_interface_type, server_object_interface_version, server_object_id AND
server_object_name
CALL nameserver.bind WITH server_object_name AND server_object_aor

SET server_object_instance TO INSTANCE OF transaction_client SERVER OBJECT
SET server_object_id TO UNIQUE INTEGER
SET server_object_host TO "myserver.mydomain.com"
SET server_object_port TO "1234"
SET server_object_interface_type TO "core::transaction_client"
SET server_object_interface_version TO = 5.1
SET server_object_name TO "fds/searchservice/myserver.mydomain.com_13280"
SET server_object_aor TO server_object_host, server_object_port,
server_object_interface_type, server_object_interface_version, server_object_id AND
server_object_name
CALL nameserver.bind WITH server_object_name AND server_object_aor
```

4.1.2 Protocol Client Messages

```
SET server_object_name TO "fds/searchservice/myserver.mydomain.com_13280"
SET server_object_type TO "core::transaction_client"
SET server_object_version TO = 5.1
CALL nameserver.resolve WITH server_object_name, server_object_type AND server_object_version
RETURNING transaction_client_proxy

SET server_object_name TO "fds/searchservice/myserver.mydomain.com_13280"
SET server_object_type TO "searchservice::configuration"
SET server_object_version TO = 5.1
CALL nameserver.resolve WITH server_object_name, server_object_type AND server_object_version
RETURNING configuration_proxy
```

```
SET transaction_id TO 0
CALL configuration_proxy.prepare_set_query_pipeline WITH transaction_id
CALL transaction_client_proxy.commit WITH transaction_id
```

5 Security

5.1 Security Considerations for Implementers

None.

5.2 Index of Security Parameters

None.

6 Appendix A: Full IDL

```
module interfaces {
  module core {
    exception unable_to_prepare_exception {
    };
    interface transaction_client {
      #pragma version transaction_client 5.1
      void commit(in long id);
      void abort(in long id);
    };
  };
  module searchservice {
    interface configuration {
      #pragma version configuration 5.1
      void prepare_set_query_pipeline(in long transid)
        raises (core::unable_to_prepare_exception);
      void prepare_set_result_pipeline(in long transid)
        raises (core::unable_to_prepare_exception);
    };
  };
};
```

7 Appendix B: 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® FAST™ Search Server 2010

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.

8 Change Tracking

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

9 Index

T

V