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](https://www.microsoft.com/en-us/technet/about/fsv3ca.htm) or the [Community Promise](https://www.microsoft.com/en-us/technet/about/fsv3ca.htm). 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](mailto: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>11/06/2009</td>
<td>0.1</td>
<td>Major</td>
<td>Initial Availability</td>
</tr>
<tr>
<td>02/19/2010</td>
<td>1.0</td>
<td>Editorial</td>
<td>Revised and edited 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>Editorial</td>
<td>Changed language and formatting in 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** ............................................................................................................. 6
   1.1 Glossary .................................................................................. 6
1.2 References ................................................................................. 7
   1.2.1 Normative References ...................................................... 7
   1.2.2 Informative References ..................................................... 7
   1.3 Protocol Overview (Synopsis) ................................................ 8
   1.4 Relationship to Other Protocols ........................................... 8
   1.5 Prerequisites/Preconditions ................................................. 9
   1.6 Applicability Statement ...................................................... 9
   1.7 Versioning and Capability Negotiation ................................ 9
   1.8 Vendor-Extensible Fields .................................................... 9
   1.9 Standards Assignments ...................................................... 9

2 **Messages** .................................................................................................................. 10
   2.1 Transport ............................................................................. 10
   2.2 Message Syntax .................................................................... 10
   2.2.1 Basic Types ....................................................................... 10
       2.2.1.1 int ......................................................................... 10
       2.2.1.2 long ....................................................................... 10
       2.2.1.3 double ................................................................. 10
       2.2.1.4 string ..................................................................... 10
       2.2.1.5 timestamp ........................................................... 10
       2.2.2 Complex Types .......................................................... 10
       2.2.2.1 cresult ................................................................. 10
       2.2.2.2 hostport ............................................................... 11
       2.2.2.3 contentdest ......................................................... 11
       2.2.2.4 dictionary ............................................................ 11
       2.2.3 Simple Types .............................................................. 11
       2.2.3.1 URI Skip Codes .................................................... 11
       2.2.3.2 Web Document Skip Codes .................................. 12
       2.2.3.3 Crawl Collection Integer Status Codes ............... 13
       2.2.3.4 Crawl Collection String Status Codes ............... 13
       2.2.3.5 Log Levels .......................................................... 13
       2.2.3.6 Crawl Mode String Status Codes ....................... 14
   2.4 Statistics Structure ................................................................. 14
   2.4.1 Statistics Dictionary ....................................................... 15
   2.4.2 Crawl Collections Dictionary ........................................ 15
   2.4.3 Statistics tuple ............................................................... 16
   2.4.4 Statistics Delta ............................................................... 16
   2.4.5 Statistics Cycles ............................................................. 16
   2.4.6 Statistics Data Types ..................................................... 17
       2.4.6.1 Continuous ......................................................... 17
       2.4.6.2 Discrete / Multi-discrete ....................................... 17
       2.4.6.3 Histogram .......................................................... 17
       2.4.6.4 Multi-dimension ................................................... 18
   2.4.7 Schema ........................................................................... 18
   2.4.8 Global Statistics Schema ............................................... 19
   2.4.9 The Crawl Site Statistics Schema ................................. 20
   2.4.10 Crawl Collection Statistics Schema ......................... 21
   2.4.11 Flattened Crawl collection Statistics Structure ........... 23

---

[MS-FSCADM] — v20120630
Crawler Administration and Status Protocol Specification

Copyright © 2012 Microsoft Corporation.

Release: July 16, 2012
3 Protocol Details

3.1 Common Details .................................................................................. 37
3.2 Client Details ...................................................................................... 37
  3.2.1 Abstract Data Model ................................................................. 37
  3.2.2 Timers ....................................................................................... 37
  3.2.3 Initialization .............................................................................. 37
  3.2.4 Higher-Layer Triggered Events .................................................. 37
  3.2.5 Message Processing Events and Sequencing Rules .................... 37
  3.2.6 Timer Events ............................................................................ 37
  3.2.7 Other Local Events ................................................................... 37
3.3 Server Details ..................................................................................... 37
  3.3.1 Abstract Data Model ................................................................. 37
  3.3.2 Timers ....................................................................................... 38

[MS-FSCADM] — v20120630
Crawler Administration and Status Protocol Specification

Copyright © 2012 Microsoft Corporation.

Release: July 16, 2012
1 Introduction

This document specifies the Crawler Administration and Status Protocol. This protocol operates between a protocol client and a Web crawler server or server farm. The protocol enables the protocol client to make requests of the Web crawler, as well as query the Web crawler for status information. This protocol is a pure client/server protocol, where the administration process is the protocol client and the Web crawler process is the protocol server.

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]:

- attribute
- checksum
- Coordinated Universal Time (UTC)
- Domain Name System (DNS)
- file system
- fully qualified domain name (FQDN)
- Hypertext Transfer Protocol (HTTP)
- Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS)
- Internet Protocol version 6 (IPv6)
- IPv4 address in string format
- IPv6 address in string format
- path
- Transmission Control Protocol (TCP)
- UTF-8
- XML

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

- absolute URI
- base port
- connection
- crawl collection
- crawl limit
- crawl queue
- crawl refetch
- crawl refresh cycle
- crawl site
- crawl statistics cycle
- crawl subcollection
- document
- file
- File Transfer Protocol (FTP)
- hyperlink
- MIME type
- multinode scheduler
- node
- node scheduler
- RSS channel
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.


1.2.2 Informative References

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

1.3 Protocol Overview (Synopsis)

This protocol uses XML to encode methods and responses and uses HTTP as a transport mechanism. It provides a means for a protocol client to issue a set of administrative and status calls to a Web crawler application.

Communication occurs when the protocol client sends an XML-encapsulated request to the protocol server. The protocol server sends a similarly encoded XML response to the protocol client. The protocol server does not initiate communication with the protocol client. The protocol client is aware of the hostname and port of the protocol server for Web crawler requests. The crawler uses the configuration that is described in the Crawler Configuration File Format Specification [MS-FSCCFG].

![Diagram of communication overview for this protocol](image)

Figure 1: Communication overview for this protocol

The Web crawler supports two distinct protocol server roles. The first role is the node scheduler role, where the Web crawler manages the current node only. The second role is the multinode scheduler role, where the Web crawler manages multiple Web crawler nodes. The supported protocol methods and behaviors are dependent on which one of the two roles is active.

1.4 Relationship to Other Protocols

This protocol uses the [XML-RPC] protocol to format requests and responses. It transmits these messages using the HTTP protocol as described in [RFC2616]. The following diagram shows the relationship of this protocol to other protocols:

![Diagram of relationship to other protocols](image)
1.5 Prerequisites/Preconditions

This protocol assumes that the protocol client has obtained the host name and port of the protocol server prior to issuing a request.

1.6 Applicability Statement

None.

1.7 Versioning and Capability Negotiation

None.

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.
2 Messages

2.1 Transport

This protocol uses the transport protocol specified in [XML-RPC].

2.2 Message Syntax

The format of the XML body requests and responses is specified in [MS-FSXTAPI]. The HTTP POST path, as specified in [RFC2616], MUST be "/RPC2".

2.2.1 Basic Types

The following sections specify the basic types used within this protocol.

2.2.1.1 int

An int is a 32-bit signed integer.

2.2.1.2 long

A long is a 64-bit signed integer.

2.2.1.3 double

A double is a double-point number that MUST adhere to the standard specified in [IEEE754]. Only signed values within a double-precision 64-bit range are supported.

2.2.1.4 string

A string is a UTF-8 encoded string literal.

2.2.1.5 timestamp

A timestamp is an int or double and MUST specify time in seconds since 1970-01-01 00:00:00 UTC.

2.2.2 Complex Types

The following sections specify the complex types used by this protocol.

2.2.2.1 cresult

A cresult is an array that contains exactly two elements. It specifies a return value of success or failure for methods that require it.

When a method finishes successfully, it sets the first element to an integer value of 1. If a method fails, it sets the first element to less than 1. The second element is a string that contains a textual description of the result of the method.

See section 4.2 for an example of how a cresult array is encoded in an [XML-RPC] response.
2.2.2.2  hostport

A hostport encapsulates a host name and port number into a string.

The format of the string MUST be <hostname>:<port>. The hostname MUST be a hostname, a fully qualified domain name (FQDN), an IPv4 address in string format, or an IPv6 address in string format. If the hostname is an Internet Protocol version 6 (IPv6) address, the hostname MUST be contained in brackets, for example, "[:]". The specified hostname MUST resolve successfully on the local node.

2.2.2.3  contentdest

A contentdest represents the destination for a content submission. If a contentdest is an empty string, the protocol server uses the default configuration. Otherwise, a contentdest is either the name of a content submission destination as specified in [MS-FSCCFG] section 2.2.4.9 or it is the string "default:<content collection>" where <content collection> specifies a valid content collection name.

2.2.2.4  dictionary

A dictionary structure is composed of key-value pairs, where each key maps to exactly one value. The keys MUST be a string data type, and the values MUST be one of the following data types: int, long, double, string, or dictionary.

In an [XML-RPC] request or response, a dictionary MUST be encoded as a structure that contains members. The dictionary keys and values are specified by the name and value XML tags that are specified in the associated data type.

See section 4.3, for an example of how a nested dictionary is encoded as a structure.

2.2.3  Simple Types

2.2.3.1  URI Skip Codes

URI skip codes specify the reason that the protocol server discarded a URI immediately after extracting it from a web document, without adding it to the crawl queue. The URI skip code is a two-letter string that is specified in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>nf</td>
<td>No follow: The URI was tagged as &quot;NoFollow&quot; by an HTML META tag as specified by [HTML], or was rejected because of the RSS channel crawl configuration settings, as specified in [MS-FSCCFG] section 2.2.4.30.</td>
</tr>
<tr>
<td>ch</td>
<td>Scheme: The URI scheme of the URI was not included in the schemes configured by the crawl configuration, as specified in [MS-FSCCFG].</td>
</tr>
<tr>
<td>ur</td>
<td>URI exclusion: The URI is not a URI to be crawled, as specified within the URI crawl rules in the crawl configuration. For more details, see [MS-FSCCFG].</td>
</tr>
<tr>
<td>ro</td>
<td>Robots disallow: The URI was excluded by a Disallow robots.txt directive, as specified in [ROBOTSTXT].</td>
</tr>
<tr>
<td>do</td>
<td>Hostname exclusion: The URI is not within the list of hostnames to crawl, as specified in the hostname crawl rules in the crawl configuration. For more details, see [MS-FSCCFG].</td>
</tr>
<tr>
<td>Value</td>
<td>Meaning</td>
</tr>
<tr>
<td>-------</td>
<td>---------</td>
</tr>
<tr>
<td>ic</td>
<td><strong>Cache exclusion</strong>: The URI was present in an internal cache; therefore it has already been added on the crawl queue.</td>
</tr>
<tr>
<td>fo</td>
<td><strong>Out of focus</strong>: The crawl configuration specified focused crawling and the URI was outside the focus. For more details, see [MS-FSCCFG] section 2.2.4.19.</td>
</tr>
<tr>
<td>de</td>
<td><strong>Depth exceeded</strong>: The crawl configuration specifies a limited depth crawl and the URI exceeded this depth. For more details, see [MS-FSCCFG] section 2.2.4.28.</td>
</tr>
</tbody>
</table>

### 2.2.3.2 Web Document Skip Codes

These specify the reason a web document is discarded after downloading it from the remote **Web server**. The skip code values are specified in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>he</td>
<td><strong>Header error</strong>: The web document contains an HTTP header that is not valid.</td>
</tr>
<tr>
<td>mi</td>
<td><strong>MIME type</strong>: The web document MIME type does not match any of the MIME types specified in the crawl configuration. For more details, see [MS-FSCCFG].</td>
</tr>
<tr>
<td>ti</td>
<td><strong>Timeout</strong>: Download of the web document exceeded the timeout as specified in the crawl configuration. For more details, see [MS-FSCCFG].</td>
</tr>
<tr>
<td>tl</td>
<td><strong>Too large</strong>: The web document exceeded the maximum size as specified in the crawl configuration. For more details, see [MS-FSCCFG].</td>
</tr>
<tr>
<td>hx</td>
<td><strong>Header excluded</strong>: The Web server sent an HTTP header that contains a header line that is excluded by the crawl configuration. For more details, see [MS-FSCCFG].</td>
</tr>
<tr>
<td>en</td>
<td><strong>Web document encoding</strong>: The encoding header specified in the HTTP header was not valid or was unsupported by the Web crawler. For more details, see [MS-FSCCFG].</td>
</tr>
<tr>
<td>cu</td>
<td><strong>Chunk decode error</strong>: The web document encoding was chunked, as specified in [RFC2616], but the Web crawler could not be decoded it.</td>
</tr>
<tr>
<td>nr</td>
<td><strong>No redirection target</strong>: The HTTP header contains an HTTP redirect, as specified in [RFC2616], but no redirect URI could be found or correctly parsed.</td>
</tr>
<tr>
<td>ni</td>
<td><strong>No Index</strong>: The web document contained a &quot;NoIndex&quot; HTML META tag as specified by [HTML] and MUST NOT be indexed.</td>
</tr>
<tr>
<td>cs</td>
<td><strong>Duplicate</strong>: The web document is identical to a previously crawled web document with a different URI.</td>
</tr>
<tr>
<td>fs</td>
<td><strong>Out of focus</strong>: The crawl configuration specifies focused crawling and the web document is outside the focus. For more details on focused crawling, see [MS-FSCCFG] section 2.2.4.19.</td>
</tr>
<tr>
<td>in</td>
<td><strong>Incomplete</strong>: The web document was only partially downloaded.</td>
</tr>
<tr>
<td>co</td>
<td><strong>Connect failure</strong>: A socket connection to the Web server was not established.</td>
</tr>
<tr>
<td>ct</td>
<td><strong>Connect timeout</strong>: A socket connection timed out.</td>
</tr>
<tr>
<td>ne</td>
<td><strong>Network error</strong>: A network error occurred.</td>
</tr>
<tr>
<td>em</td>
<td><strong>Empty web document</strong>: The Web server sent an empty web document.</td>
</tr>
<tr>
<td>Value</td>
<td>Meaning</td>
</tr>
<tr>
<td>-------</td>
<td>---------</td>
</tr>
<tr>
<td>eh</td>
<td>Empty HTTP header: The Web server sent an empty HTTP header.</td>
</tr>
<tr>
<td>rs</td>
<td>RSS No Index: The web document is an RSS channel feed and the crawl configuration specifies that RSS channels are not indexed separately. For more details, see [MS-FSCCFG] section 2.2.4.30.</td>
</tr>
<tr>
<td>ot</td>
<td>Other error: Error not specified by one of the previous categories.</td>
</tr>
</tbody>
</table>

### 2.2.3.3 Crawl Collection Integer Status Codes

These specify the state of crawl collections. The status code MUST be of type int and specified in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The crawl collection is either crawling or idle.</td>
</tr>
<tr>
<td>2</td>
<td>Reserved, MUST NOT be used.</td>
</tr>
<tr>
<td>3</td>
<td>The Web crawler is deleting the crawl collection.</td>
</tr>
<tr>
<td>4</td>
<td>Crawl collection is suspended.</td>
</tr>
<tr>
<td>5</td>
<td>Crawl collection is suspended because of the lack of free disk space on the local file system.</td>
</tr>
<tr>
<td>6</td>
<td>Crawl collection executing regular cleanup routine.</td>
</tr>
<tr>
<td>7</td>
<td>Reserved, MUST NOT be used.</td>
</tr>
<tr>
<td>8</td>
<td>Crawl collection is suspended because of a variable delay configuration. For details, see [MS-FSCCFG] section 2.2.4.23.</td>
</tr>
</tbody>
</table>

### 2.2.3.4 Crawl Collection String Status Codes

These specify the state of crawl collections. The status code MUST be of type string and specified in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>crawling</td>
<td>The crawl collection is crawling.</td>
</tr>
<tr>
<td>suspended</td>
<td>The crawl collection is suspended. Crawling MUST NOT occur.</td>
</tr>
<tr>
<td>compacting</td>
<td>The Web crawler is performing disk storage cleanup. Crawling MUST NOT occur.</td>
</tr>
<tr>
<td>zombie</td>
<td>The crawl collection is being deleted. Crawling MUST NOT occur while the deletion is processing. A new crawl collection with the same name cannot be created while the delete process is executing.</td>
</tr>
</tbody>
</table>

### 2.2.3.5 Log Levels

Log level codes MUST be specified as one of the bitmasks described in the following table.
### 2.2.3.6 Crawl Mode String Status Codes

These specify the crawl mode of the Web crawler when a crawl limit has been reached. The limits are determined by the crawl configuration, as specified in [MS-FSCCFG] section 2.2.4.18.

The status code MUST be of type string and specified in the following table. If no crawl limit has been reached this value MUST be the empty string.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Refreshing(limit=max_doc)</td>
<td>The maximum web document count has been reached.</td>
</tr>
<tr>
<td>Refreshing(limit=disk_free)</td>
<td>The amount of disk free space is less than the minimum.</td>
</tr>
<tr>
<td>Refreshing(limit=user)</td>
<td>The refreshing mode has been manually engaged, for example by the CollectionEnableRefreshOnly method in section 2.2.10.2.</td>
</tr>
</tbody>
</table>

### 2.2.4 Statistics Structure

The following figure specifies the data structure that the protocol server sends by using the CollectionGetStatistics method that is specified in section 2.2.10.5. Both the "dspec" and "global" keys MUST be present.
The CollectionGetSiteStatistics method specified in section 2.2.10.4 sends a similar structure where crawl collection MUST be substituted with the crawl site and the "global" key MUST NOT be present.

### 2.2.4.1 Statistics Dictionary

The Statistics dictionary is a dictionary with keys of data type string, whose contents are specified in the following table.

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>dspec</td>
<td>dictionary</td>
<td>A dictionary of crawl configuration options, see [MS-FSCCFG], that the protocol client uses to calculate progress information. See section 2.2.2.4.</td>
</tr>
<tr>
<td>global</td>
<td>tuple</td>
<td>Global statistics.</td>
</tr>
<tr>
<td>&lt;crawl collection name&gt;</td>
<td>Tuple</td>
<td>Crawl collection statistics.</td>
</tr>
</tbody>
</table>

### 2.2.4.2 Crawl Collections Dictionary

The Crawl collections dictionary MUST contain one or more string keys that are equivalent to the names of crawl collections that are associated with the Web crawler. Each key maps to a dictionary that specifies configuration options for that crawl collection.

This dictionary MUST contain keys of data type int that are converted to data type string, as specified in the following table.
### Name | Meaning
---|---
refresh | Specifies the **crawl refresh cycle** length, in minutes, for the crawl collection.
refresh_mode | Specifies the refresh mode. MUST be a value from the following table. For more details about the refresh modes, see [MS-FSCCFG].

The following table specifies the valid values for the "refresh_mode" key in this **dictionary**.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Append refresh mode.</td>
</tr>
<tr>
<td>2</td>
<td>Prepend Refresh mode.</td>
</tr>
<tr>
<td>3</td>
<td>Reserved.</td>
</tr>
<tr>
<td>4</td>
<td>Scratch refresh mode.</td>
</tr>
<tr>
<td>5-7</td>
<td>Reserved.</td>
</tr>
<tr>
<td>8</td>
<td>Soft refresh mode.</td>
</tr>
<tr>
<td>9-31</td>
<td>Reserved.</td>
</tr>
<tr>
<td>32</td>
<td>Adaptive refresh mode.</td>
</tr>
</tbody>
</table>

#### 2.2.4.3 Statistics tuple

The **statistics tuple** consists of the **crawl statistics delta** and the **crawl statistics cycles**. The **crawl statistics delta** specifies any statistics updates that are not yet merged into the appropriate **statistics cycle** structures. The crawl statistics cycles is an array that contains statistics for the last k crawl refresh cycles, in addition to the cycle indicated by the key value of "cOmPlEtE", which tracks the entire crawl. The value of k MUST be an int value that is greater than 0. The cycle specified by the "cOmPlEtE" key MUST contain statistics aggregated for all cycles.

#### 2.2.4.4 Statistics Delta

The crawl statistics delta MUST be implemented as one of three schemas. It is a global schema, a crawl collection schema, or a crawl site schema. It MUST contain only statistics that are aggregated over the last 1-2 minutes and have not yet been merged into a crawl statistics cycle. See section **2.2.4.5**, for more details. When a delta is merged into a cycle, all its values MUST be re-initialized to 0. The **__starts__ and __stopts__ attributes** are used to track the beginning and ending times of the statistics delta. See section **2.2.4.7** for more details.

#### 2.2.4.5 Statistics Cycles

The crawl statistics cycles are in a **dictionary** that contains the most recent statistics. A crawl statistics cycle corresponds to a crawl refresh cycle.

The keys in the **dictionary** are associated with the number of the crawl refresh cycle, which MUST be of type int and MUST begin at 0 and increase by 1 for each cycle. This means that the highest number will be the most recent crawl statistics cycle.
The key "cOmPlEtE" MUST be used to specify the crawl statistics cycle that covers all cycles. The complete crawl statistics cycle contains the sum of all aggregated statistics cycles, including the current cycle, even if the cycles are no longer present in the statistics dictionary.

The values in the dictionary MUST be of type global schema, crawl collection schema or crawl site schema.

### 2.2.4.6 Statistics Data Types

The statistics data types are a set of composite data types that encapsulate four basic data types `int`, `long`, `double`, and string. Numeric values are dynamically created as data type `int` or `long` depending on the size of the value. The `double` data type is used to store large numeric values because it is natively supported as specified in [XML-RPC]. The protocol client MUST support all four data types. The actual data type that the protocol MUST use to transport these structures is based on the data marshal format that is specified in [MS-FSWCU].

#### 2.2.4.6.1 Continuous

The `continuous` data type is the data type for a continuous variable, suitable for graphing. It MUST be encoded as an array of length four, as specified in the following table. Each element within the table is of data type `int`, `long`, or `double`.

<table>
<thead>
<tr>
<th>Array index</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>The current value of the variable.</td>
</tr>
<tr>
<td>1</td>
<td>The minimum value of the variable.</td>
</tr>
<tr>
<td>2</td>
<td>The maximum value of the variable.</td>
</tr>
<tr>
<td>3</td>
<td>The number of times the variable has been updated.</td>
</tr>
</tbody>
</table>

#### 2.2.4.6.2 Discrete / Multi-discrete

The `discrete` data type encapsulates values that are represented independently of earlier samples. The data type MUST be encoded as an array of length 4, as specified in section 2.2.4.6.1.

The `multidiscrete` data type represents a value that MUST be a sum of individual `discrete` values. These values are not contained in the data type. For this protocol, the `multidiscrete` data type MUST be identical to the `discrete` data type.

#### 2.2.4.6.3 Histogram

The `histogram` data type represents a set of values for a frequency distribution bar graph. The protocol MUST associate each numeric value with its number of occurrences. This data type MUST be encoded as an array of length 4, as specified by the following table.

<table>
<thead>
<tr>
<th>Array index</th>
<th>Data type</th>
<th>Meaning</th>
</tr>
</thead>
</table>
| 0           | dictionary | `Dictionary` representing the histogram. In the key-value pairs, the keys specify the data values, and the values specify the occurrences of each data value.  
`Dictionary` keys are of type string. `Dictionary` values are of type `int`, `long` or `double`. |
<table>
<thead>
<tr>
<th>Array index</th>
<th>Data type</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>string</td>
<td>MUST contain the value &quot;None&quot;.</td>
</tr>
<tr>
<td>2</td>
<td>string</td>
<td>MUST contain the value &quot;None&quot;.</td>
</tr>
<tr>
<td>3</td>
<td>int, long, or double</td>
<td>The number of times the variable has been updated.</td>
</tr>
</tbody>
</table>

### 2.2.4.6.4 Multi-dimension

The multidimension data type specifies a two-dimensional array that contains attributes that are associated with a set of values. This data type MUST be encoded as an array of length four, as specified by the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Data type</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>dictionary</td>
<td>Dictionary that represents the two dimensional array. Keys MUST be of type string, and specify the first dimension. Values MUST be of type array, and specify the second dimension. Each array MUST contain a sequence of 0 or more values of data type int, long, or double.</td>
</tr>
<tr>
<td>1</td>
<td>string</td>
<td>Reserved, MUST contain the value &quot;None&quot;.</td>
</tr>
<tr>
<td>2</td>
<td>string</td>
<td>Reserved, MUST contain the value &quot;None&quot;.</td>
</tr>
<tr>
<td>3</td>
<td>int, long, or double</td>
<td>The number of times the variable has been updated.</td>
</tr>
</tbody>
</table>

### 2.2.4.7 Schema

This specifies the base class schema for the statistics, as specified in the following table. It consists of a dictionary whose keys are specified as lower case strings, and whose values specify the schema type, metadata, and contents. The mandatory keys MUST be included in the dictionary.

<table>
<thead>
<tr>
<th>Name</th>
<th>Mandatory</th>
<th>Data type</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>schemaname</strong></td>
<td>Yes</td>
<td>string</td>
<td>Specifies the schema type, which MUST be one of &quot;globalschema&quot;, &quot;siteschema&quot;, or &quot;collectionschema&quot;.</td>
</tr>
<tr>
<td><strong>startts</strong></td>
<td>No</td>
<td>timestamp</td>
<td>Specifies when statistics aggregation was started.</td>
</tr>
<tr>
<td><strong>stopts</strong></td>
<td>No</td>
<td>timestamp</td>
<td>Specifies when statistics aggregation was stopped.</td>
</tr>
<tr>
<td>name</td>
<td>Yes</td>
<td>string</td>
<td>See the following sections.</td>
</tr>
<tr>
<td>discrete</td>
<td>No</td>
<td>array</td>
<td>Specifies an array that contains discrete statistics attributes.</td>
</tr>
<tr>
<td>continuous</td>
<td>No</td>
<td>array</td>
<td>Specifies an array that contains continuous values.</td>
</tr>
<tr>
<td>histogram</td>
<td>No</td>
<td>array</td>
<td>Specifies an array that contains histogram values.</td>
</tr>
<tr>
<td>multidimension</td>
<td>No</td>
<td>array</td>
<td>Specifies an array that contains multidimension values.</td>
</tr>
<tr>
<td>Name</td>
<td>Mandatory</td>
<td>Data type</td>
<td>Meaning</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------</td>
<td>-----------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>multidiscrete</td>
<td>No</td>
<td>array</td>
<td>See discrete.</td>
</tr>
</tbody>
</table>

The contents of the `discrete`, `continuous`, `histogram`, `multidimension`, and `multidiscrete` keys MUST be an array of two elements as specified in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Type</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>string</td>
<td>The name of the statistics attribute. This value MUST be a valid name based on the current schema and attribute type.</td>
</tr>
<tr>
<td>1</td>
<td>array</td>
<td>The statistics value, which MUST be specified according to the definition of the attribute type. The attribute type MUST be one of <code>discrete</code>, <code>continuous</code>, <code>histogram</code>, <code>multidimension</code>, or <code>multidiscrete</code>.</td>
</tr>
</tbody>
</table>

### 2.2.4.8 Global Statistics Schema

The global statistics schema contains statistics that is not specific to any specific crawl collection, and MUST be formatted as specified in section 2.2.4.7. The `__schemaname__` MUST contain the value "globalschema". The attributes and their contents are specified in the following table.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Aggregation type</th>
<th>Data type</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uptime</td>
<td>Discrete</td>
<td><code>timestamp</code></td>
<td>The time that the protocol server began recording statistics which SHOULD be the time when the Web crawler began running.</td>
</tr>
<tr>
<td>DNSWrite</td>
<td>Continuous</td>
<td><code>int</code>, <code>long</code>, or <code>double</code></td>
<td>Specifies, in bytes, the amount of Domain Name System (DNS) traffic that the protocol server has sent to a DNS server.</td>
</tr>
<tr>
<td>DNSRead</td>
<td>Continuous</td>
<td><code>int</code>, <code>long</code>, or <code>double</code></td>
<td>Specifies, in bytes, the amount of DNS traffic from a DNS server.</td>
</tr>
<tr>
<td>DNSRequests</td>
<td>Continuous</td>
<td><code>int</code>, <code>long</code>, or <code>double</code></td>
<td>Specifies the number of DNS requests that the protocol server has sent to a DNS server.</td>
</tr>
<tr>
<td>DNSResponses</td>
<td>Continuous</td>
<td><code>int</code>, <code>long</code>, or <code>double</code></td>
<td>Specifies the number of DNS responses that the protocol server has received from a DNS server.</td>
</tr>
<tr>
<td>DNSRetries</td>
<td>Continuous</td>
<td><code>int</code>, <code>long</code>, or <code>double</code></td>
<td>Specifies the number of DNS requests that the protocol server has sent to a DNS server as retries.</td>
</tr>
<tr>
<td>DNSTimeout</td>
<td>Continuous</td>
<td><code>int</code>, <code>long</code>, or <code>double</code></td>
<td>Specifies the number of DNS requests that resulted in a timeout.</td>
</tr>
<tr>
<td>DNSResponse</td>
<td>Histogram</td>
<td>string</td>
<td>A histogram of DNS responses from a DNS server. These are specified as a number expressed as a string, and MUST be one of the DNS RCODE values that are specified in [RFC1035].</td>
</tr>
<tr>
<td>DNSRateLimit</td>
<td>Discrete</td>
<td><code>int</code>, <code>long</code>, or <code>double</code></td>
<td>Specifies the maximum number of DNS requests that the protocol server has sent, in requests per second.</td>
</tr>
</tbody>
</table>
## 2.2.4.9 The Crawl Site Statistics Schema

The crawl site statistics are differentiated at a per-crawl-site level, and they MUST be formatted as specified in section 2.2.4.7. The \_schemaname\_ variable MUST contain the value "siteschema". The attributes and their contents are of data type \textbf{int}, \textbf{long}, or \textbf{double}, and MUST correspond to the values in the following table.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Aggregation type</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processed</td>
<td>Continuous</td>
<td>Number of URIs extracted from the crawl queue and requested from the Web server during the crawl process.</td>
</tr>
<tr>
<td>Downloaded</td>
<td>Continuous</td>
<td>Number of HTTP responses that the Web server sent during crawling.</td>
</tr>
<tr>
<td>Stored</td>
<td>Continuous</td>
<td>Number of web documents, both modified and new, that were written to the web document storage during crawling.</td>
</tr>
<tr>
<td>Modified</td>
<td>Continuous</td>
<td>Number of modified web documents that were encountered during crawling. The Web crawler MUST track each URI that was downloaded and determine whether subsequent web documents from the same URI are the same identical web document or a modified version.</td>
</tr>
<tr>
<td>Deleted</td>
<td>Continuous</td>
<td>Number of web documents that were deleted from the Web crawler web document storage during crawling.</td>
</tr>
<tr>
<td>Unchanged</td>
<td>Continuous</td>
<td>Number of unchanged web documents that were encountered during crawling. A web document is unchanged if its computed checksum is the same as the value computed when the same URI was crawled in a previous crawl refresh cycle.</td>
</tr>
<tr>
<td>DBUpdates</td>
<td>Continuous</td>
<td>Number of URI metadata write processes.</td>
</tr>
<tr>
<td>ReadNet</td>
<td>Continuous</td>
<td>Number of bytes received on HTTP, \textbf{HTTPS} and \textbf{File Transfer Protocol (FTP)} network connections. This value MUST include information only from network connections that request URIs during crawling. Crawler inter process communication MUST NOT be included.</td>
</tr>
<tr>
<td>WriteNet</td>
<td>Continuous</td>
<td>Number of bytes sent on HTTP, HTTPS and FTP network connections. Crawler interprocess communication MUST NOT be included in this number.</td>
</tr>
<tr>
<td>HasLastMod</td>
<td>Continuous</td>
<td>Number of web documents that were downloaded during crawling that contain a &quot;Last-Modified&quot; HTTP header, as specified in \textbf{[RFC2616]}.</td>
</tr>
<tr>
<td>HasETag</td>
<td>Continuous</td>
<td>Number of web documents that were downloaded during crawling that contain an &quot;ETag&quot; HTTP header, as specified in \textbf{[RFC2616]}.</td>
</tr>
<tr>
<td>DLTime</td>
<td>Continuous</td>
<td>Total accumulated download time, in seconds, for all web documents that were crawled.</td>
</tr>
<tr>
<td>DocSize</td>
<td>Continuous</td>
<td>Total size, in bytes, of all web documents that were crawled.</td>
</tr>
<tr>
<td>Retries</td>
<td>Continuous</td>
<td>Number of URIs that failed crawling and resulted or will result in a retry to download the same URI.</td>
</tr>
<tr>
<td>Attribute</td>
<td>Aggregation type</td>
<td>Meaning</td>
</tr>
<tr>
<td>--------------------</td>
<td>------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>HasSitemap</td>
<td>Continuous</td>
<td>Number of robots.txt files that were downloaded and that contain one or more &quot;Sitemap:&quot; directives that represent the URI of sitemaps, as specified in [SITEMAPS].</td>
</tr>
<tr>
<td>NodeSchedulerFwd</td>
<td>Continuous</td>
<td>Number of URIs that were forwarded between crawler nodes in a multinode installation.</td>
</tr>
<tr>
<td>LocalFwd</td>
<td>Continuous</td>
<td>Number of URIs that are hyperlinks to other crawl sites and were not forwarded to a different crawler node.</td>
</tr>
<tr>
<td>Local</td>
<td>Continuous</td>
<td>Number of URIs that are local hyperlinks within a crawl site.</td>
</tr>
<tr>
<td>LastUri</td>
<td>Discrete</td>
<td>The most recent URI crawled.</td>
</tr>
<tr>
<td>QueueLength</td>
<td>Discrete</td>
<td>The number of URIs on the Web crawler’s URI queue.</td>
</tr>
<tr>
<td>CrawlStarted</td>
<td>Discrete</td>
<td>A timestamp that specifies the time when the crawl site began crawling. This value MUST be updated whenever the Web crawler begins a new crawl refresh cycle on the crawl site. The value MUST be updated whenever the Web crawler begins crawling the crawl site after a Web crawler restart.</td>
</tr>
<tr>
<td>CrawlEnded</td>
<td>Discrete</td>
<td>Specifies the time when the crawl site stopped crawling.</td>
</tr>
<tr>
<td>Epoch</td>
<td>Discrete</td>
<td>The crawl refresh cycle number. This value MUST be increased by 1 whenever the Web crawler begins crawling a crawl site.</td>
</tr>
<tr>
<td>LastRefresh</td>
<td>Discrete</td>
<td>A timestamp that specifies the time when the crawl site most recently began crawling.</td>
</tr>
<tr>
<td>HTTPResponse</td>
<td>Histogram</td>
<td>Specifies a histogram of HTTP response codes that were sent to the Web crawler, as specified by [RFC2616]. HTTP response codes MUST be specified as a number expressed as a string.</td>
</tr>
<tr>
<td>URISkip</td>
<td>Histogram</td>
<td>Specifies a histogram of URI skip codes. A skipped URI is one that was extracted from a web document, but not added to the crawl queue or crawled for some specific reason. URI skip codes MUST be one of the values specified in section 2.2.3.1.</td>
</tr>
<tr>
<td>DocSkip</td>
<td>Histogram</td>
<td>Specifies a histogram of web document skip codes. A skipped web document was downloaded from the remote Web server, but was not stored or sent to the indexing engine. Web document skip codes are specified in section 2.2.3.2.</td>
</tr>
<tr>
<td>MimeType</td>
<td>Histogram</td>
<td>Specifies a histogram of web document content types that were encountered in HTTP Content-Type headers during crawling. MUST be as specified in [RFC2616].</td>
</tr>
</tbody>
</table>

### 2.2.4.10 Crawl Collection Statistics Schema

The crawl collection statistics schema MUST specify crawl collection specific statistics level and MUST use the definition specified in section 2.2.4.7. The __schemaname__ MUST be specified as the string "collectionschema".

The attributes and their contents are specified in section 2.2.4.9 and in the following table. Section 2.2.4.9 applies to the entire crawl collection, rather than any specific crawl site.
<table>
<thead>
<tr>
<th>Attribute</th>
<th>Aggregation type</th>
<th>Data type</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uptime</td>
<td>Discrete</td>
<td>timestamp</td>
<td>Specifies the timestamp when the crawler began recording statistics for this crawl collection. This MUST correspond to the time when the crawl collection was added to the Web crawler.</td>
</tr>
<tr>
<td>ActiveSites</td>
<td>Discrete</td>
<td>int, long, or double</td>
<td>Specifies the number of crawl sites currently being crawled. This MUST be 0 if the Web crawler is idle.</td>
</tr>
<tr>
<td>Feeding</td>
<td>Discrete</td>
<td>int</td>
<td>Specifies whether the Web crawler is currently submitting Web documents to the indexing engine. This MUST be 0 if content submission is paused, 1 otherwise.</td>
</tr>
<tr>
<td>Status</td>
<td>Discrete</td>
<td>int, long, or double</td>
<td>Specifies the crawl collection status of the Web crawler. The value MUST be as specified in section 2.2.3.3.</td>
</tr>
<tr>
<td>CrawlMode</td>
<td>Discrete</td>
<td>string</td>
<td>If a crawl limit has been reached, the value of this string MUST be specified according to section 2.2.3.6.</td>
</tr>
<tr>
<td>StartUriStatus</td>
<td>Multidimension</td>
<td>string</td>
<td>Specifies the status of URIs being added on the crawl queue from start URI files, as specified by [MS-FSCCFG]. If start URIs are not currently being added to the crawl queue, then the structure MUST be empty. Otherwise, the structure MUST associate the start URI files to the status strings. The status strings are of the format &quot;&lt;filesize&gt; &lt;fileposition&gt;&quot;, where &lt;filesize&gt; MUST specify the total file size in bytes and &lt;fileposition&gt; MUST specify the current file pointer position in bytes.</td>
</tr>
<tr>
<td>FeedingDestinations</td>
<td>Multidimension</td>
<td>int, long, or double</td>
<td>Specifies the content collections that are associated with this crawl collection, and also specifies whether the process that submits site content to the indexing engine is currently active or paused. Each dictionary key MUST be associated with a content destination in the crawl configuration as specified in [MS-FSCCFG] section 2.2.4.9. If content submission is paused, the associated dictionary value MUST have a string value of 0; otherwise, the string MUST have a value of 1. The structure MUST be empty if there are no content destinations configured for the crawl collection that are not the default destination.</td>
</tr>
<tr>
<td>SumActiveSites</td>
<td>Multidiscrete</td>
<td>int, long, or double</td>
<td>Multinode scheduler behavior: The attribute MUST specify the sum of the ActiveSites attribute across all node schedulers. Single node scheduler behavior: The attribute</td>
</tr>
<tr>
<td>Attribute</td>
<td>Aggregation type</td>
<td>Data type</td>
<td>Meaning</td>
</tr>
<tr>
<td>--------------</td>
<td>------------------</td>
<td>--------------------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>PPFailed</td>
<td>Continuous</td>
<td>int, long, or double</td>
<td>Specifies the number of URIs that failed submission to the indexing system.</td>
</tr>
<tr>
<td>PPSucceeded</td>
<td>Continuous</td>
<td>int, long, or double</td>
<td>The number of URIs that were successfully indexed by the indexing system.</td>
</tr>
<tr>
<td>PPFed</td>
<td>Continuous</td>
<td>int, long, or double</td>
<td>The number of URIs that were successfully submitted to the indexing system.</td>
</tr>
<tr>
<td>PPDeleted</td>
<td>Continuous</td>
<td>int, long, or double</td>
<td>The number of URIs that were successfully deleted from the indexing system.</td>
</tr>
<tr>
<td>PPAdded</td>
<td>Continuous</td>
<td>int, long, or double</td>
<td>The number of URIs that were processed by the Web crawler and that were previously unseen and have unique web document checksums.</td>
</tr>
<tr>
<td>PPModified</td>
<td>Continuous</td>
<td>int, long, or double</td>
<td>The number of URIs that were processed by the Web crawler and that were previously seen and have unique web document checksums.</td>
</tr>
<tr>
<td>PPURLsChange</td>
<td>Continuous</td>
<td>int, long, or double</td>
<td>The number of urischange_operations, as specified in [MS-FSCF] section 2.2.42, that were sent to the indexing engine.</td>
</tr>
<tr>
<td>PPPartialUpdate</td>
<td>Continuous</td>
<td>int, long, or double</td>
<td>Reserved, MUST be set to 0 or 0.0.</td>
</tr>
<tr>
<td>PPChecksums</td>
<td>Discrete</td>
<td>int, long, or double</td>
<td>The number of unique web document checksums that exist in the Web crawler store.</td>
</tr>
<tr>
<td>SumPPChecksums</td>
<td>Multidiscrete</td>
<td>int, long, or double</td>
<td>Multinode scheduler behavior: The attribute MUST specify the sum of the PPChecksums attribute across all node schedulers. Node scheduler behavior: The attribute MUST be 0.</td>
</tr>
</tbody>
</table>

### 2.2.5 Flattened Crawl collection Statistics Structure

This structure is used to encapsulate crawl collection statistics in a dictionary, which MUST correspond to the specification in the following table.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Data type</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>ActiveSites</td>
<td>int</td>
<td>See section 2.2.4.10.</td>
</tr>
<tr>
<td>Processed</td>
<td>double</td>
<td>See section 2.2.4.9.</td>
</tr>
<tr>
<td>Downloaded</td>
<td>double</td>
<td>See section 2.2.4.9.</td>
</tr>
<tr>
<td>Modified</td>
<td>double</td>
<td>See section 2.2.4.9.</td>
</tr>
<tr>
<td>Attribute</td>
<td>Data type</td>
<td>Meaning</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Stored</td>
<td>double</td>
<td>See section 2.2.4.9.</td>
</tr>
<tr>
<td>Deleted</td>
<td>double</td>
<td>See section 2.2.4.9.</td>
</tr>
<tr>
<td>ReadNet</td>
<td>double</td>
<td>See section 2.2.4.9.</td>
</tr>
<tr>
<td>WriteNet</td>
<td>double</td>
<td>See section 2.2.4.9.</td>
</tr>
<tr>
<td>DocSize</td>
<td>double</td>
<td>Total aggregated web document size.</td>
</tr>
<tr>
<td>DocSizeAvg</td>
<td>double</td>
<td>Average web document size in bytes.</td>
</tr>
<tr>
<td>DocSizeMax</td>
<td>double</td>
<td>Maximum web document size in bytes.</td>
</tr>
<tr>
<td>Retries</td>
<td>double</td>
<td>See section 2.2.4.9.</td>
</tr>
<tr>
<td>NodeSchedulerFwd</td>
<td>double</td>
<td>See section 2.2.4.9.</td>
</tr>
<tr>
<td>LocalFwd</td>
<td>double</td>
<td>See section 2.2.4.9.</td>
</tr>
<tr>
<td>Local</td>
<td>double</td>
<td>See section 2.2.4.9.</td>
</tr>
<tr>
<td>Epoch</td>
<td>int</td>
<td>See section 2.2.4.9.</td>
</tr>
<tr>
<td>LastRefresh</td>
<td>int</td>
<td>See section 2.2.4.9.</td>
</tr>
<tr>
<td>Feeding</td>
<td>int</td>
<td>See section 2.2.4.10.</td>
</tr>
<tr>
<td>CrawlMode</td>
<td>string</td>
<td>See section 2.2.4.10.</td>
</tr>
<tr>
<td>MimeType</td>
<td>dictionary</td>
<td>A histogram of web document content types that were encountered during crawling, as specified in [RFC2616]. Dictionary keys MUST be the MIME types, and their corresponding values MUST be the frequency.</td>
</tr>
<tr>
<td>HTTPResponse</td>
<td>dictionary</td>
<td>Specifies a histogram of HTTP response codes that were sent to the Web crawler, as specified by [RFC2696]. HTTP response codes MUST be a string literal that contains the HTTP response code number. Dictionary keys MUST be the HTTP response codes, and their corresponding values MUST be the frequency.</td>
</tr>
<tr>
<td>Uptime</td>
<td>double</td>
<td>See section 2.2.4.10.</td>
</tr>
<tr>
<td>DLTime</td>
<td>double</td>
<td>See section 2.2.4.9.</td>
</tr>
<tr>
<td>DLTimeAvg</td>
<td>double</td>
<td>Average time, in seconds, that is spent downloading a web document.</td>
</tr>
<tr>
<td>DLTimeMax</td>
<td>double</td>
<td>Maximum time, in seconds, that is spent downloading a web document.</td>
</tr>
<tr>
<td>Status</td>
<td>string</td>
<td>Crawling status. MUST be one of the following strings: &quot;Crawling&quot;, &quot;Zombie&quot;, &quot;Suspended&quot;, &quot;Disk full&quot;, &quot;Compacting&quot;, and &quot;Variable delay suspended&quot;.</td>
</tr>
<tr>
<td>URISkip</td>
<td>dictionary</td>
<td>See section 2.2.4.9.</td>
</tr>
<tr>
<td>Attribute</td>
<td>Data type</td>
<td>Meaning</td>
</tr>
<tr>
<td>---------------</td>
<td>-----------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>DocSkip</td>
<td>dictionary</td>
<td>See section 2.2.4.9.</td>
</tr>
<tr>
<td>DocumentStore</td>
<td>int</td>
<td>Estimated number of web documents with unique URIs that reside in the Web crawler store. MUST be calculated using the following formula: DocumentStore = Stored - Modified - Deleted</td>
</tr>
<tr>
<td>Progress</td>
<td>array</td>
<td>Craw refresh cycle progress. MUST be an array of two elements. The first element specifies the completion percentage of the current crawl refresh cycle. The second element is a human readable string that contains the amount of time until the next refresh occurs.</td>
</tr>
<tr>
<td>FirstUpdate</td>
<td>double</td>
<td>A timestamp that specifies when the crawl collection started crawling. This value MUST be updated whenever the Web crawler begins crawling the crawl site. The value MUST be updated when the Web crawler begins a new crawl after a crawler restart.</td>
</tr>
<tr>
<td>StatUpdate</td>
<td>double</td>
<td>Specifies the time when the crawl collection ended the crawl.</td>
</tr>
<tr>
<td>DocRate</td>
<td>double</td>
<td>Average web document download rate specified as web documents per second.</td>
</tr>
<tr>
<td>DataRateIn</td>
<td>double</td>
<td>Average incoming data rate from protocol headers and web document data for the sites that are being crawled. MUST be specified as bytes per second.</td>
</tr>
<tr>
<td>DataRateOut</td>
<td>double</td>
<td>Average outgoing data rate for information such as HTTP request messages. MUST be specified as bytes per second.</td>
</tr>
<tr>
<td>PPFailed</td>
<td>double</td>
<td>See section 2.2.4.10.</td>
</tr>
<tr>
<td>PPSucceeded</td>
<td>double</td>
<td>See section 2.2.4.10.</td>
</tr>
<tr>
<td>PPFed</td>
<td>double</td>
<td>See section 2.2.4.10.</td>
</tr>
<tr>
<td>PPDeleted</td>
<td>double</td>
<td>See section 2.2.4.10.</td>
</tr>
<tr>
<td>PPAdded</td>
<td>double</td>
<td>See section 2.2.4.10.</td>
</tr>
<tr>
<td>PPModified</td>
<td>double</td>
<td>See section 2.2.4.10.</td>
</tr>
<tr>
<td>PPUURLsChange</td>
<td>double</td>
<td>See section 2.2.4.10.</td>
</tr>
<tr>
<td>PPChecksums</td>
<td>double</td>
<td>See section 2.2.4.10.</td>
</tr>
</tbody>
</table>

2.2.6 Flattened Global Statistics Structure

This structure is used to encapsulate global crawl statistics in a dictionary, which MUST correspond to the specification in the following table.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Data type</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>DNSWrite</td>
<td>double</td>
<td>See section 2.2.4.8.</td>
</tr>
<tr>
<td>DNSRead</td>
<td>double</td>
<td>See section 2.2.4.8.</td>
</tr>
<tr>
<td>Attribute</td>
<td>Data type</td>
<td>Meaning</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------</td>
<td>-------------------------------------</td>
</tr>
<tr>
<td>DNSRequest</td>
<td>double</td>
<td>See section 2.2.4.8.</td>
</tr>
<tr>
<td>DNSResponse</td>
<td>double</td>
<td>See section 2.2.4.8.</td>
</tr>
<tr>
<td>DNSResponses</td>
<td>dictionary</td>
<td>See section 2.2.4.8.</td>
</tr>
<tr>
<td>DNSRetries</td>
<td>double</td>
<td>See section 2.2.4.8.</td>
</tr>
<tr>
<td>DNSTimeout</td>
<td>double</td>
<td>See section 2.2.4.8.</td>
</tr>
<tr>
<td>DNSRateLimit</td>
<td>double</td>
<td>See section 2.2.4.8.</td>
</tr>
<tr>
<td>DNSRate</td>
<td>double</td>
<td>Number of sent DNS requests per second.</td>
</tr>
<tr>
<td>DNSDataRateIn</td>
<td>double</td>
<td>Number of DNS traffic bytes received per second.</td>
</tr>
<tr>
<td>DNSDataRateOut</td>
<td>double</td>
<td>Number of DNS traffic bytes sent per second.</td>
</tr>
</tbody>
</table>

### 2.2.7 Error Handling

The protocol supports a special fault response message for reporting errors to the protocol client, as specified in [XML-RPC]. The message consists of a fault code whose value MUST be 1 and a fault string that specifies the error that occurred.

Most errors that occur in the methods of this specification generate faults when an error occurs. When a fault is generated, it replaces the return value of the method.

### 2.2.8 Web crawler Management

All methods specified in this section MUST have all arguments specified.

#### 2.2.8.1 GetLogLevel

The GetLogLevel method queries the protocol server for its current log verbosity level.

```c
int GetLogLevel()
```

**Return value:** MUST return the log level, which MUST be a mask of one or more of the log levels specified in section 2.2.3.5.

#### 2.2.8.2 GetNodeSchedulerXmlRpcPorts

The GetNodeSchedulerXmlRpcPorts method queries the protocol server for a list of known node schedulers, as follows.

```c
array GetNodeSchedulerXmlRpcPorts()
```

**Return value:** If the protocol server is a multinode scheduler, then the return value MUST specify an array with at least 1 element. The array elements MUST be arrays that are two elements long, where the first element specifies a hostname as a string and the second specifies a port as an int. The combination of a hostname and port MUST uniquely identify a node scheduler that is registered with the multinode scheduler.
If the protocol server is not a multinode scheduler, the protocol server MUST send a fault as specified in [XML-RPC].

2.2.8.3  GetSiteManagerNum

The **GetSiteManagerNum** method queries the protocol server for the number of site managers that the crawler node controls. A site manager is a service that runs on the crawler node and manages crawling for a set of crawl sites, as follows.

```c
int GetSiteManagerNum()
```

**Return value:** MUST return the number of site managers that the crawler node controls. If the protocol server is a crawler multinode scheduler, the number of site managers MUST be 0.

2.2.8.4  IsDistributed

The **IsDistributed** method queries the protocol server for the multinode Web crawler setup state, as follows.

```c
int IsDistributed()
```

**Return value:** MUST return a value of 1 if the protocol server is part of a multinode Web crawler setup, regardless of the role of the protocol server. Otherwise, it MUST return a value of 0.

2.2.8.5  IsMultiNodeScheduler

The **IsMultiNodeScheduler** method queries the protocol server whether it is a multinode scheduler or a node scheduler, as follows.

```c
int IsMultiNodeScheduler()
```

**Return value:** MUST return 1 if the protocol server represents a multinode scheduler in a multinode Web crawler setup. If the protocol server is a node scheduler the value returned MUST be 0.

2.2.8.6  SetLogLevel

The **SetLogLevel** method sets the log verbosity level for the protocol server, as follows.

```c
int SetLogLevel(int Level)
```

The protocol server MUST set its log verbosity to the level specified as a result of this method. If the protocol server has the role of a multinode scheduler, this method MUST also be forwarded to each individual node scheduler before the method returns the response.

**Level:** The log level. See section 2.2.3.5 for more details about log levels.

**Return value:** MUST return 1 if the method succeeds, 0 otherwise.

2.2.9  Crawl Collection Management

All of the methods that are specified in this section MUST specify all arguments.
2.2.9.1 AddURIs

The AddURIs method adds one or more URIs to the crawl queue, as follows.

```cresult AddURIs(string Collection, int Urgent, array URIs)```

**Collection**: The name of crawl collection.

**Urgent**: MUST be either 0 or 1. When set to 1 the URIs MUST be processed as soon as possible by the Web crawler.

**URIs**: An array of strings specifying URIs that are scheduled for crawling. The URIs MUST be absolute URIs.

**Return value**: The result of the method.

2.2.9.2 AddURIFile

The AddURIFile method adds one or more URIs contained in a text file to the crawl queue, as follows.

```cresult AddURIFile(string Collection, int Urgent, string URIFile)```

**Collection**: The name of crawl collection. If the crawl collection does not exist then a fault MUST be sent, as specified in [XML-RPC].

**Urgent**: MUST be either 0 or 1. When set to 1 the URIs MUST be processed as soon as possible by the Web crawler.

**URIFile**: The absolute path of a file on the local file system on the protocol server. The file MUST be in the UTF-8 encoding. Each individual URI MUST be specified on a single line. The URIs MUST be absolute.

**Return value**: The result of the method.

2.2.9.3 CollectionAdd

The CollectionAdd method adds a new crawl collection to the Web crawler, as follows.

```cresult CollectionAdd(string ConfigData, int Force)```

If the crawl collection already exists the existing configuration MUST be updated instead. The crawler MUST use the configuration specified in the Crawler Configuration File Format Specification [MS-FSCCFG].

A partial configuration can be specified. If a partial configuration is submitted and the crawl collection already exists, this method merges the partial configuration with the existing configuration, a process where all settings that are specified in both configurations MUST be overridden by the new configuration. If the crawl collection does not already exist it MUST be merged with the default configuration values as specified in [MS-FSCCFG].

**ConfigData**: A crawl configuration in XML format.
force: MUST be either 0 or 1. If the value is 1 then the configuration MUST be accepted even if the protocol server is acting as a node scheduler in a multinode setup, otherwise a fault MUST be returned. A multinode scheduler or node scheduler that performs as a single node MUST ignore this option and always add the crawl collection.

Return value: The result of the method.

2.2.9.4 CollectionDelete

The CollectionDelete method deletes an existing crawl collection from the Web crawler, as follows.

```c
}result CollectionDelete(string Collection, int Force)
```

Collection: The name of the crawl collection to be deleted.

Force: MUST be either 0 or 1. If the value is 1 then the configuration MUST be deleted even if the protocol server is acting as a node scheduler in a multinode setup, otherwise a fault MUST be returned. A multinode scheduler or node scheduler that performs as a single node MUST ignore this option and always delete the crawl collection.

Return value: The result of the method.

2.2.9.5 CollectionDeleteSite

The CollectionDeleteSite method deletes a crawl site from the specified crawl collection in the Web crawler store and from the searchable index, as follows.

```c
}result CollectionDeleteSite(string Collection, hostport Site)
```

Collection: The name of the crawl collection.

Site: The crawl site to be deleted.

Return value: The result of the method.

2.2.9.6 CollectionDeleteURIs

The CollectionDeleteURIs method deletes one or more URIs from the specified crawl collection in the Web crawler store and from the searchable index, as follows.

```c
}result CollectionDeleteURIs(string Collection, array URIs)
```

Collection: The name of the crawl collection.

URIs: An array of strings that specifies one or more URIs to be deleted. The URIs specified MUST be absolute.

Return value: The result of the method.

2.2.9.7 CollectionDeleteURIFile

The CollectionDeleteURIFile method deletes one or more URIs specified in a file from the specified crawl collection in the Web crawler store and from the searchable index, as follows.

```c
}result CollectionDeleteURIFile(string Collection, file URIFile)
```
2.2.9.8 CollectionGetConfigurationXML

The CollectionGetConfigurationXML method returns the specified crawl configuration as XML, as follows.

```csharp
string CollectionGetConfigurationXML(string Collection)
```

**Collection:** The name of the crawl collection.

**Return value:** The crawl configuration in XML, as specified in [MS-FSCCFG].

2.2.9.9 CollectionGetList

The CollectionGetList method returns the list of crawl collections currently configured in the Web crawler, as follows.

```csharp
array CollectionGetList()
```

**Return value:** An array of strings, each of which specifies the name of a crawl collection.

2.2.9.10 CollectionGetSiteRoutingAddress

The CollectionGetSiteRoutingAddress method returns information about which node scheduler is responsible for the specified crawl site in a multinode Web crawler setup, as follows.

```csharp
cresult CollectionGetSiteRoutingAddress(string Collection, hostport Site)
```

**Collection:** The name of the crawl collection.

**Site:** The crawl site for which to retrieve routing information.

**Return value:** On success the second element in the cresult is an array of two elements that specify the node scheduler to which to route the crawl site. The first element is the hostname and the second element is the port of the node scheduler on that node.

2.2.9.11 CollectionPreemptSite

The CollectionPreemptSite method stops the crawling that is being performed on the specified crawl site, as follows.

```csharp
cresult CollectionPreemptSite(string Collection, hostport Site)
```
The protocol server SHOULD stop crawling immediately, but MUST resume the crawling at a later time.

**Collection**: The name of the crawl collection.

**Site**: The crawl site to be preempted.

**Return value**: The result of the method.

### 2.2.9.12 **CollectionQuarantineSite**

The **CollectionQuarantineSite** method suspends crawling of the specified crawl site for the specified time interval. The protocol server SHOULD stop crawling immediately, but MUST resume the crawling at a later time.

```c
result CollectionQuarantineSite(string Collection, hostport Site, int Time)
```

Any URIs that belong to the quarantined site found during quarantine MUST be added on a separate queue for the duration of the quarantine.

**Collection**: The name of the crawl collection.

**Site**: The crawl site to be preempted.

**Time**: The duration of the quarantine in seconds.

**Return value**: The result of the method.

### 2.2.9.13 **CollectionRefetch**

The **CollectionRefetch** method schedules a crawl refetch of the specified crawl collection, as follows.

```c
result CollectionRefetch(string Collection, string SubCollection)
```

**Collection**: The name of the crawl collection.

**SubCollection**: The name of a crawl subcollection, as specified in [MS-FSCCFG]. This option MUST be an empty string, in which case the entire crawl collection MUST be re-crawled, or the refetch process MUST only apply to the specified crawl subcollection.

**Return value**: The result of the method.

### 2.2.9.14 **CollectionRefetchURI**

The **CollectionRefetchURI** method schedules a crawl refetch process for the specified URI or crawl site, as follows.

```c
result CollectionRefetchURI(string Collection, string URI, int Refeed, int Site, int Immediately)
```

**Collection**: The name of the crawl collection.

**URI**: The URI to retrieved. MUST specify an absolute URI.
Refeed: MUST be either 0 or 1. When the value is 1, the URI MUST also be sent to the processing pipeline, regardless of whether it was modified after the last time it was retrieved.

Site: MUST be either 0 or 1. When the value is 1 the method MUST re-crawl the entire crawl site associated with the URI, not just the URI itself.

Immediately: MUST be either 0 or 1. When the value is 1 the URI MUST be treated as an urgent URI, meaning the protocol server SHOULD crawl the URI as soon as possible.

Return value: The result of the method.

2.2.9.15 CollectionReprocessSite

The CollectionReprocessSite method sends all known web documents from the specified crawl site to the indexing engine for indexing, as follows.

```cresult CollectionReprocessSite(string Collection, contentdest Destination, hostport Site)```

Collection: The name of the crawl collection.

Destination: The content submission destination.

Site: The crawl site to be processed.

Return value: The result of the method.

2.2.9.16 CollectionReprocessSitePrefix

The CollectionReprocessSitePrefix method sends all known web documents from the specified crawl site matching the URI prefix to the indexing engine for indexing, as follows.

```cresult CollectionReprocessSitePrefix(string Collection, hostport Site, string Prefix, contentdest Destination)```

Collection: The name of the crawl collection.

Site: The crawl site to be processed.

Prefix: A URI prefix which MUST be used to perform a prefix match on each URI considered a candidate for processing.

Destination: The content submission destination.

Return value: The result of the method.

2.2.9.17 CollectionReprocessURI

The CollectionReprocessURI method sends all known web documents from the specified crawl site to the indexing engine for indexing, as follows.

```cresult CollectionReprocessURI(string Collection, contentdest Destination, array URIs)```

Collection: The name of the crawl collection.
Destination: The content submission destination.

URIs: An array of strings that specifies one or more URIs to be processed. The URIs specified MUST be absolute URIs. The protocol server SHOULD continue processing URIs even if non-valid URIs are encountered.

Return value: The result of the method.

2.2.9.18 CollectionResume

The CollectionResume method resumes crawling of a crawl collection that was previously suspended, for example by using a call to the CollectionSuspend method as specified in section 2.2.9.20, as follows.

```cresult CollectionResume(string Collection)```

If the collection is already suspended, the protocol server MUST send a fault as specified in [XML-RPC].

Collection: The name of the crawl collection.

Return value: The result of the method.

2.2.9.19 CollectionResumeFeeding

The CollectionResumeFeeding method resumes submission of crawled content to the indexing engine for a crawl collection that was previously suspended, for example by using a call to the CollectionSuspendFeeding method as specified in section 2.2.9.21, as follows.

```cresult CollectionResumeFeeding(string Collection, contentdest Destination)```

Collection: The name of the crawl collection.

Destination: The content submission destination.

Return value: The result of the method.

2.2.9.20 CollectionSuspend

The CollectionSuspend method suspends crawling of a crawl collection. Crawling of the crawl collection MUST NOT already be suspended, as follows.

```cresult CollectionSuspend(string Collection)```

Collection: The name of the crawl collection.

Return value: The result of the method.

2.2.9.21 CollectionSuspendFeeding

The CollectionSuspendFeeding method suspends content submissions of crawled content for indexing. The crawl collection MUST NOT already have web document submission set to suspended, as follows.

cresult CollectionSuspendFeeding(string Collection, contentdest Destination)

**Collection**: The name of the crawl collection.

**Destination**: The content submission destination.

**Return value**: The result of the method.

### 2.2.10 Crawl Collection Status

All methods specified in this section MUST have all arguments specified.

#### 2.2.10.1 CollectionDisableRefreshOnly

The `CollectionDisableRefreshOnly` method changes from crawling only previously visited content to also crawling new content, as follows.

```csharp
CollectionDisableRefreshOnly(string Collection)
```

The protocol server MUST now download new content as well as revisiting old content.

**Collection**: The name of the crawl collection.

**Return value**: The result of the method.

#### 2.2.10.2 CollectionEnableRefreshOnly

The `CollectionEnableRefreshOnly` method only revisits previously crawled content only, as follows.

```csharp
CollectionEnableRefreshOnly(string Collection)
```

The Web crawler MUST only revisit previously crawled content, and MUST NOT download any URIs not already present in the Web crawler store.

**Collection**: The name of the crawl collection.

**Return value**: The result of the method.

#### 2.2.10.3 CollectionGetStatus

The `CollectionGetStatus` method returns the status of the Web crawler, as follows.

```csharp
CollectionGetStatus(string Collection)
```

**Collection**: The name of the crawl collection.

**Return value**: The current state of the crawl collection, formatted as specified in section 2.2.3.3.
2.2.10.4 CollectionGetSiteStatistics

The CollectionGetSiteStatistics method MUST return the aggregated statistics for the specified crawl site, as follows.

```
struct CollectionGetSiteStatistics(string Collection, string Site)
```

**Collection:** MUST specify a valid crawl collection name.

**Site:** MUST specify a valid crawl site crawled by the crawl collection.

**Return value:** The method MUST return a valid statistics structure as specified in section 2.2.3.5. The statistics structure MUST contain statistics for the specified crawl site and the structure MUST be serialized using the pyfastmarshal format, as specified in [MS-FSWCU].

2.2.10.5 CollectionGetStatistics

The CollectionGetStatistics method MUST return the aggregated statistics for the specified crawl collection. The method MUST also return global statistics. If no crawl collections are specified the method MUST return statistics for all configured crawl collections, as follows.

```
struct CollectionGetStatistics(string Collection)
```

**Collection:** MUST specify a valid crawl collection name or an empty string.

**Return value:** The method MUST return a valid statistics structure as specified in section 2.2.4. The statistics structure MUST contain statistics for the specified crawl collection as well as global statistics. The structure MUST be marshalled using the pyfastmarshal format, as specified in [MS-FSWCU].

2.2.10.6 CollectionGetStatistics2

The CollectionGetStatistics2 method MUST return the aggregated statistics for the specified crawl collection, as follows.

```
array CollectionGetStatistics2(string Collection)
```

**Collection:** MUST specify a valid crawl collection name.

**Return value:** The method MUST return an array of two elements. The array MUST be as specified in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Specifies the result of the method. Its value MUST be 1 or less than 1. If the int has a value of 1, the method was successful and the other element of the array contains a dictionary type. Otherwise, the method was unsuccessful and the other element of the array contains a string that specifies the error text.</td>
</tr>
<tr>
<td>1</td>
<td>Contains a string or a dictionary data type. If this method returns a dictionary data type, it MUST contain one, two, or three keys of type string. The valid values are &quot;cur&quot;, &quot;prev&quot; and &quot;complete&quot;, each of which refers to a dictionary as specified in section 2.2.4. The contents of the dictionaries MUST be one of the following:</td>
</tr>
</tbody>
</table>

[MS-FSCADM] — v20120630
Crawler Administration and Status Protocol Specification

Copyright © 2012 Microsoft Corporation.

Release: July 16, 2012
### Value | Meaning
--- | ---
| The "cur" **dictionary** data type refers to a **dictionary** that MUST contain statistics from the current crawl refresh cycle. |  
| The "prev" **dictionary** data type refers to a **dictionary** that MUST contain statistics from the previous crawl refresh cycle. If there is no previous cycle available this key MUST NOT be returned. |  
| The "complete" **dictionary** data type refers to a **dictionary** that MUST contain statistics for the entire lifetime of the crawl collection. |  

The **dictionary** referred to by the key MUST be empty when no statistics are available.

### 2.2.10.7 GetGlobalStatistics

The **GetGlobalStatistics** method MUST return the aggregated global statistics for the Web crawler, as follows.

```c
array GetGlobalStatistics()
```

**Return value:** The method MUST return an array of two elements. The array MUST be as specified in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Specifies The result of the method. Its value MUST be 1 or less than 1. If the int has a value of 1, the method was successful and the other element of the array contains a <strong>dictionary</strong> data type. Otherwise, the method was unsuccessful and the other element of the array contains a string that specifies the error text.</td>
</tr>
</tbody>
</table>
| 1 | Contains a string or a **dictionary** data type. If this value contains a **dictionary** data type, it MUST contain one or two keys of type string. The valid values are "delta" and "complete", each of which refers to a **dictionary** as specified in section 2.2.4.8. The contents of the dictionaries MUST be one of the following:  
| The "delta" **dictionary** type refers to a **dictionary** that MUST contain recently aggregated statistics over a short time window. |  
| The "complete" **dictionary** data type refers to a **dictionary** that MUST contain statistics for the entire lifetime of the Web crawler. |  

The **dictionary** to which the key refers MUST be empty when no statistics are available.
3 Protocol Details

3.1 Common Details

None.

3.2 Client Details

3.2.1 Abstract Data Model

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

3.2.2 Timers

None.

3.2.3 Initialization

The protocol requires setup of a Transmission Control Protocol (TCP) connection between protocol client and protocol server. The port numbers used for the connection is base port. The protocol client MUST initiate the connection.

3.2.4 Higher-Layer Triggered Events

None.

3.2.5 Message Processing Events and Sequencing Rules

None.

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

3.3 Server Details

In its initial state, the Web crawler MUST have no configured crawl collections. Methods specified in section 2 are used to alter the state of the Web crawler, by adding or removing crawl collections. The protocol server MUST persist these altered states.

3.3.1 Abstract Data Model

The protocol server MUST listen for incoming connections, process incoming XML-RPC requests and respond to them in a timely manner. If an unexpected error occurs during processing the protocol server MUST send a fault as specified in [XML-RPC].
The protocol server MUST persist the following information:

**Crawl collection configurations:** A data structure that contains information associated with crawl collections that are currently known to the Web crawler, in addition to their respective configuration settings. This data MUST be persisted to disk so the Web crawler can restart and resume crawling.

**Statistics:** Statistics MUST be updated during run time and MUST also be persisted to disk between restarts. The current and global crawl statistics cycles MUST always be retained. Statistics for recent crawl refresh cycles MUST also be retained, but the number of cycles to retain is implementation dependent.

**Crawl queues:** Individual crawl queues for each crawl collection MUST be persisted to disk. Quarantined URIs MUST be persisted to different queues, one or more per crawl collection.

**Current activity:** The protocol server MUST keep track of which crawl sites are being crawled at any time, as well as the state of crawls being performed on each crawl collection. The state of the crawl collection MUST also be persisted to disk between restarts.

**Crawl store:** The protocol server MUST maintain a persistent Web crawler store on disk. The minimum information that MUST be stored in the Web crawler store is the fields that specify which URIs have been crawled, the checksum of each URI, and when the URI was last crawled.

### 3.3.2 Timers

None.

### 3.3.3 Initialization

The protocol server MUST start its XML-RPC server implementation as soon as it can process incoming requests. The protocol server MUST listen on the base port. See section 3.3.3 for more details about protocol initialization.

The protocol server MUST register with the Configuration Service as specified in [MS-FSCX], and implement the following methods that are required by that protocol: `ConfigurationChanged`, `ReRegister` and `ping`.

### 3.3.4 Higher-Layer Triggered Events

None.

### 3.3.5 Message Processing Events and Sequencing Rules

Specific processing rules apply to the following methods:

**CollectionAdd:** This method MUST update the crawl configuration state when the new crawl collection is added. An empty statistics structure MUST be associated with the crawl collection. The crawl collection state MUST be set to a value of "running".

**CollectionDelete:** This method MUST initiate deletion of all crawl collection specific data structures, including crawl configuration, statistics, crawl queues, Web crawler store and other internal structures. While the delete process is in progress, the status code of the crawl collection MUST be set to a value of "zombie" until the collection has been completely removed. The Web crawler MUST remove any remaining references associated with the crawl collection from its structures so the crawl collection can be added as a new crawl collection.
3.3.6 Timer Events

None.

3.3.7 Other Local Events

None.
4 Protocol Examples

These examples contain only the XML body for each XML-RPC message. See [XML-RPC] for examples of HTTP headers.

4.1 CollectionGetList

This example shows a **CollectionGetList** method, which returns the crawl collections known to the Web crawler as an array of strings. In this example, the crawl collections are "contoso" and "web".

Request

```xml
<?xml version='1.0'?>
<methodCall>
  <methodName>CollectionGetList</methodName>
  <params>
  </params>
</methodCall>
```

Response

```xml
<?xml version='1.0'?>
<methodResponse>
  <params>
    <param>
      <value>
        <array>
          <data>
            <value><string>contoso</string></value>
            <value><string>web</string></value>
          </data>
        </array>
      </value>
    </param>
  </params>
</methodResponse>
```

4.2 AddURIs

In this example, the **AddURIs** method adds two URIs to the crawl queue for the crawl collection "contoso". The result value is in the **cresult** format, and consists of an array with two elements; those two elements are a return code and a description of the results.

Request

```xml
<?xml version='1.0'?>
<methodCall>
  <methodName>AddURIs</methodName>
  <params>
    <param>
      <value><string>contoso</string></value>
    </param>
    <param>
      <value><int>0</int></value>
    </param>
  </params>
</methodCall>
```
<param>
  <value>
    <array>
      <data>
        <value><string>http://www.contoso.com/</string></value>
        <value><string>http://www.contoso.com/about/</string></value>
      </data>
    </array>
  </value>
</param>

Response

<?xml version='1.0'?>
<methodResponse>
  <params>
    <param>
      <value>
        <array><data>
          <value><int>1</int></value>
          <value><string>Queued collection contoso with 2 URIs</string></value>
        </data>
      </value>
    </param>
  </params>
</methodResponse>

4.3 CollectionGetStatistics2

In this example, a result is returned from the CollectionGetStatistics2 method. It demonstrates nested structures, and a variety of data types.

Request

<?xml version='1.0'?>
<methodCall>
  <methodName>CollectionGetStatistics2</methodName>
  <params>
    <param>
      <value><string>contoso</string></value>
    </param>
  </params>
</methodCall>

Response

<?xml version='1.0'?>
<methodResponse>
  <params>
    <param>
      <value>
        <array><data>
        </data>
      </value>
    </param>
  </params>
</methodResponse>
<value><int>1</int></value>
<value><struct>
  <member>
    <name>cur</name>
    <value><struct>
      <member>
        <name>PPAdded</name>
        <value><double>40.0</double></value>
      </member>
      <member>
        <name>ReadNet</name>
        <value><double>174367.0</double></value>
      </member>
      <member>
        <name>Feeding</name>
        <value><int>1</int></value>
      </member>
      <member>
        <name>DLTimeMax</name>
        <value><double>0.20227789878845215</double></value>
      </member>
      <member>
        <name>Retries</name>
        <value><double>0.0</double></value>
      </member>
      <member>
        <name>DocSizeAvg</name>
        <value><double>4071.0</double></value>
      </member>
      <member>
        <name>PFURLsChange</name>
        <value><double>0.0</double></value>
      </member>
      <member>
        <name>DocumentStore</name>
        <value><int>40</int></value>
      </member>
      <member>
        <name>URISkip</name>
        <value><struct>
          </struct></value>
      </member>
      <member>
        <name>Uptime</name>
        <value><double>1234197854.7414169</double></value>
      </member>
      <member>
        <name>Epoch</name>
        <value><int>0</int></value>
      </member>
      <member>
        <name>FirstUpdate</name>
        <value><double>1234195119.4230039</double></value>
      </member>
      <member>
        <name>Progress</name>
        <value><array><data>
          <value><int>2</int></value>
          <value><string>1 days 8h 34m 23s until scratch refresh</string></value>
        </array></value>
      </member>
    </struct></value>
  </member>
</struct></value>
<data>
  <member>
    <name>Local</name>
    <value><double>320.0</double></value>
  </member>
  <member>
    <name>Status</name>
    <value><string>Crawling</string></value>
  </member>
  <member>
    <name>PPSucceeded</name>
    <value><double>0.0</double></value>
  </member>
  <member>
    <name>PPDeleted</name>
    <value><double>0.0</double></value>
  </member>
  <member>
    <name>Deleted</name>
    <value><double>0.0</double></value>
  </member>
  <member>
    <name>LocalFwd</name>
    <value><double>0.0</double></value>
  </member>
  <member>
    <name>DataRateIn</name>
    <value><double>63.722122866030894</double></value>
  </member>
  <member>
    <name>PPModified</name>
    <value><double>0.0</double></value>
  </member>
  <member>
    <name>Downloaded</name>
    <value><double>40.0</double></value>
  </member>
  <member>
    <name>DLTimeAvg</name>
    <value><double>0.0075536489486694334</double></value>
  </member>
  <member>
    <name>WriteNet</name>
    <value><double>12790.0</double></value>
  </member>
  <member>
    <name>HTTPResponse</name>
    <value><struct>
      <member>
        <name>200</name>
        <value><double>41.0</double></value>
      </member>
      <member>
        <name>404</name>
        <value><double>1.0</double></value>
      </member>
    </struct></value>
  </member>
</data>
<member>
  <name>DLTime</name>
  <value><double>0.30214595794677734</double></value>
</member>

<member>
  <name>MimeType</name>
  <value><struct>
    <member>
      <name>text/html</name>
      <value><double>40.0</double></value>
    </member>
  </struct></value>
</member>

<member>
  <name>Modified</name>
  <value><double>0.0</double></value>
</member>

<member>
  <name>DocSkip</name>
  <value><struct></struct></value>
</member>

<member>
  <name>ActiveSites</name>
  <value><int>0</int></value>
</member>

<member>
  <name>DocRate</name>
  <value><double>0.014617931802699111</double></value>
</member>

<member>
  <name>Processed</name>
  <value><double>40.0</double></value>
</member>

<member>
  <name>DocSizeMax</name>
  <value><double>8460.0</double></value>
</member>

<member>
  <name>StatUpdated</name>
  <value><string>26.2s</string></value>
</member>

<member>
  <name>NodeSchedulerFwd</name>
  <value><double>0.0</double></value>
</member>

<member>
  <name>PPChecksums</name>
  <value><double>40.0</double></value>
</member>

<member>
  <name>LastRefresh</name>
  <value><int>1234195119</int></value>
</member>

<member>
  <name>LastUpdate</name>
  <value><double>1234197855.7882121</double></value>
</member>
<name>PPFed</name>
   <value><double>0.0</double></value>
</member>

<member>
   <name>DataRateOut</name>
   <value><double>4.6740836939130403</double></value>
</member>

<member>
   <name>Unchanged</name>
   <value><double>0.0</double></value>
</member>

<member>
   <name>DocSize</name>
   <value><double>162862.0</double></value>
</member>

<member>
   <name>Stored</name>
   <value><double>40.0</double></value>
</member>

<member>
   <name>CrawlMode</name>
   <value><string></string></value>
</member>

<member>
   <name>PPPartialUpdate</name>
   <value><double>0.0</double></value>
</member>

<member>
   <name>PPFailed</name>
   <value><double>0.0</double></value>
</member>

<member>
   <name>complete</name>
   <value><struct>
      <member>
         <name>PPAdded</name>
         <value><double>40.0</double></value>
      </member>
      <member>
         <name>ReadNet</name>
         <value><double>174367.0</double></value>
      </member>
      <member>
         <name>Feeding</name>
         <value><int>1</int></value>
      </member>
      <member>
         <name>DLTimeMax</name>
         <value><double>0.20227789878845215</double></value>
      </member>
      <member>
         <name>Retries</name>
         <value><double>0.0</double></value>
      </member>
      <member>
         <name>DocSizeAvg</name>
         <value><double>4071.0</double></value>
      </member>
   </struct></value>
</member>
<name>Deleted</name>
  <value><double>0.0</double></value>
</member>

<member>
  <name>LocalFwd</name>
  <value><double>0.0</double></value>
</member>

<member>
  <name>DataRateIn</name>
  <value><double>63.722122866030894</double></value>
</member>

<member>
  <name>PPModified</name>
  <value><double>0.0</double></value>
</member>

<member>
  <name>Downloaded</name>
  <value><double>40.0</double></value>
</member>

<member>
  <name>Stored</name>
  <value><double>40.0</double></value>
</member>

<member>
  <name>DLTimeAvg</name>
  <value><double>0.0075536489486694334</double></value>
</member>

<member>
  <name>WriteNet</name>
  <value><double>12790.0</double></value>
</member>

<member>
  <name>HTTPResponse</name>
  <value><struct>
    <member>
      <name>200</name>
      <value><double>41.0</double></value>
    </member>
    <member>
      <name>404</name>
      <value><double>1.0</double></value>
    </member>
  </struct></value>
</member>

<member>
  <name>DLTime</name>
  <value><double>0.30214595794677734</double></value>
</member>

<member>
  <name>DNSTimeout</name>
  <value><double>0.0</double></value>
</member>

<member>
  <name>MimeType</name>
  <value><struct>
    <member>
      <name>text/html</name>
      <value><double>40.0</double></value>
    </member>
  </struct></value>
</member>
<struct>
  <member>
    <name>LastUpdate</name>
    <value><double>1234197855.7882121</double></value>
  </member>
  <member>
    <name>Modified</name>
    <value><double>0.0</double></value>
  </member>
  <member>
    <name>DocSkip</name>
    <value><struct>
      </struct></value>
  </member>
  <member>
    <name>ActiveSites</name>
    <value><int>0</int></value>
  </member>
  <member>
    <name>DocRate</name>
    <value><double>0.014617931802699111</double></value>
  </member>
  <member>
    <name>Processed</name>
    <value><double>40.0</double></value>
  </member>
  <member>
    <name>DocSizeMax</name>
    <value><double>8460.0</double></value>
  </member>
  <member>
    <name>StatUpdated</name>
    <value><string>26.2s</string></value>
  </member>
  <member>
    <name>NodeSchedulerFwd</name>
    <value><double>0.0</double></value>
  </member>
  <member>
    <name>PPChecksums</name>
    <value><double>40.0</double></value>
  </member>
  <member>
    <name>LastRefresh</name>
    <value><int>1234195119</int></value>
  </member>
  <member>
    <name>DNSRateLimit</name>
    <value><double>10.0</double></value>
  </member>
  <member>
    <name>PPFed</name>
    <value><double>0.0</double></value>
  </member>
  <member>
    <name>DataRateOut</name>
    <value><double>4.6740836939130403</double></value>
  </member>
</struct>
<member>
  <name>DNSRequests</name>
  <value><double>0.0</double></value>
</member>

<member>
  <name>Unchanged</name>
  <value><double>0.0</double></value>
</member>

<member>
  <name>DNSRetries</name>
  <value><double>0.0</double></value>
</member>

<member>
  <name>DocSize</name>
  <value><double>162862.0</double></value>
</member>

<member>
  <name>DNSResponses</name>
  <value><double>0.0</double></value>
</member>

<member>
  <name>CrawlMode</name>
  <value><string></string></value>
</member>

<member>
  <name>DNSRead</name>
  <value><double>0.0</double></value>
</member>

<member>
  <name>PPPartialUpdate</name>
  <value><double>0.0</double></value>
</member>

<member>
  <name>PPFailed</name>
  <value><double>0.0</double></value>
</member>

<member>
  <name>delta</name>
  <value><struct>
    <member>
      <name>DNSRateLimit</name>
      <value><double>10.0</double></value>
    </member>
    <member>
      <name>DNSDataRateOut</name>
      <value><double>0.0</double></value>
    </member>
    <member>
      <name>DNSRetries</name>
      <value><double>0.0</double></value>
    </member>
    <member>
      <name>DNSRequests</name>
      <value><double>0.0</double></value>
    </member>
    <member>
      <name>DataRateIn</name>
    </member>
  </struct></value>
</member>
<value><double>0.0</double></value>
</member>

<member>
  <name>DNSResponse</name>
  <value><struct>
    </struct></value>
</member>

<member>
  <name>DNSRate</name>
  <value><double>0.0</double></value>
</member>

<member>
  <name>DataRateOut</name>
  <value><double>0.0</double></value>
</member>

<member>
  <name>DocRate</name>
  <value><double>0.0</double></value>
</member>

<member>
  <name>DNSRead</name>
  <value><double>0.0</double></value>
</member>

<member>
  <name>DNSDataRateIn</name>
  <value><double>0.0</double></value>
</member>

<member>
  <name>DNSTimeout</name>
  <value><double>0.0</double></value>
</member>

<member>
  <name>DNSWrite</name>
  <value><double>0.0</double></value>
</member>

<member>
  <name>DNSResponses</name>
  <value><double>0.0</double></value>
</member>

</struct></value>
</member>
</member>
</data></array></value>
</param>
</params>
</methodResponse>
5 Security

5.1 Security Considerations for Implementers

None.

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® 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.
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
  client 37
  server 37
AddURIFile method 28
AddURIs method
details 28
example 40
Applicability 9

B
Base class statistics schema 18
Basic types
double 10
int 10
long 10
overview (section 2.2.1 10, section 2.2.1 10)
string 10
timestamp 10
Basic Types message 10

C
Capability negotiation 9
Change tracking 53
Client
  abstract data model 37
  higher-layer triggered events 37
  initialization 37
  message processing 37
  other local events 37
  overview 37
  sequencing rules 37
  timer events 37
  timers 37
CollectionAdd method 28
CollectionDelete method 29
CollectionDeleteSite method 29
CollectionDeleteURIFile method 29
CollectionDeleteURIs method 29
CollectionDisableRefreshOnly method 34
CollectionEnableRefreshOnly method 34
CollectionGetConfigurationXML method 30
CollectionGetList method
details 30
example 40
CollectionGetSiteRoutingAddress method 30
CollectionGetSiteStatistics method 35
CollectionGetStatistics method 35
CollectionGetStatistics2 method
details 35
example 41
CollectionGetStatus method 34
CollectionPreemptSite method 30
CollectionQuarantineSite method 31
CollectionRefresh method 31
CollectionRefreshURI method 31
CollectionReprocessSite method 32
CollectionReprocessSitePrefix method 32
CollectionResume method 33
CollectionResumeFeeding method 33
CollectionSuspend method 33
CollectionSuspendFeeding method 33
Complex types
collectiondictionarystringstatuscodesimpletype 13
Crawl Collection Management message 27
Crawl collection management methods
  AddURIFile 28
  AddURIs 28
  CollectionAdd 28
  CollectionDelete 29
  CollectionDeleteSite 29
  CollectionDeleteURIFile 29
  CollectionDeleteURIs 29
  CollectionGetConfigurationXML 30
  CollectionGetList 30
  CollectionGetSiteRoutingAddress 30
  CollectionPreemptSite 30
  CollectionQuarantineSite 31
  CollectionRefresh 31
  CollectionRefreshURI 31
  CollectionReprocessSite 32
  CollectionReprocessSitePrefix 32
  CollectionResume 33
  CollectionResumeFeeding 33
  CollectionSuspend 33
  CollectionSuspendFeeding 33
overview 27
crawl collection statistics schema 21
Crawl Collection Status message 34
Crawl collection status methods
  CollectionDisableRefreshOnly 34
  CollectionEnableRefreshOnly 34
  CollectionGetSiteStatistics 35
  CollectionGetStatistics 35
  CollectionGetStatistics2 35
  CollectionGetStatus 34
  GetGlobalStatistics 36
overview 34
crawl collection string status code simple type 13
crawl collections dictionary structure 15
crawl mode string status code simple type 14
crawl site statistics schema 20
crawl statistics cycle structure 16
crawl statistics delta structure 16
Crawler management methods
  GetLogLevel 26

54 / 57

[MS-FSCADM] — v20120630
Crawler Administration and Status Protocol Specification
Copyright © 2012 Microsoft Corporation.
Release: July 16, 2012
GetNodeSchedulerXmlRpcPort 26
GetSiteManagerNum 27
IsDistributed 27
IsMultiNodeScheduler 27
overview 26
SetLogLevel 27
cresult complex type 10

D
Data model - abstract
   client 37
   server 37
Data types - statistics structures 17
dictionary complex type 11
double basic type 10

E
Error handling - messages 26
Error Handling message 26
Examples
   AddURIs method 40
   CollectionGetList method 40
   CollectionGetStatistics2 method 41
   overview 40

F
Fields - vendor-extensible 9
flattened crawl collection statistics structure 23
Flattened Crawl collection Statistics Structure message 23
flattened global statistics structure 25
Flattened Global Statistics Structure message 25

G
GetGlobalStatistics method 36
GetLogLevel method 26
GetNodeSchedulerXmlRpcPort method 26
GetSiteManagerNum method 27
global statistics schema 19
Glossary 6

H
Higher-layer triggered events
   client 37
   server 38
   hostport complex type 11

I
Implementer - security considerations 51
Index of security parameters 51
Informative references 8
Initialization
   client 37
   server 38
   int basic type 10
   Introduction 6
   IsDistributed method 27
   IsMultiNodeScheduler method 27

L
log level simple type 13
long basic type 10

M
Message processing
   client 37
   server 38
Messages
   base class statistics schema 18
   Basic Types 10
   Complex Types 10
   Crawl Collection Management 27
crawl collection statistics schema 21
Crawl Collection Status 34
crawl collections dictionary structure 15
crawl site statistics schema 20
crawl statistics cycle structure 16
crawl statistics delta structure 16
Error Handling (section 2.2.7 26, section 2.2.7 26)
Flattened Crawl collection Statistics Structure (section 2.2.5 23, section 2.2.5 23)
Flattened Global Statistics Structure (section 2.2.6 25, section 2.2.6 25)
global statistics schema 19
Statistics data types 17
Statistics dictionary structure 15
Statistics Structure (section 2.2.4 14, section 2.2.4 14)
statistics tuple structure 16
syntax 10
transport 10
Web crawler Management 26
Messages - basic types
double 10
int 10
long 10
overview 10
string 10
timestamp 10
Messages - complex types
contentdest 11
cresult 10
dictionary 11
hostport 11
overview 10
Messages - crawl collection management
AddURIFile method 28
AddURIs method 28
CollectionAdd method 28
CollectionDelete method 29
CollectionDeleteSite method 29
CollectionDeleteURIS method 29
CollectionGetConfigurationXML method 30
CollectionGetList method 30
overview (section 3.1 37, section 3.3 37) sequencing rules 38
timer events 39
timers 38
SetLogLevel method 27
Simple types
crawl collection integer status code 13
crawl collection string status code 13
crawl mode string status code 14
log level 13
overview 11
URI skip code 11
web document skip code 12
Statistics schema
base class 18
crawl collection 21
crawl site 20
global 19
Statistics Structure message 14
Statistics structures
crawl collections dictionary 15
crawl statistics cycle 16
overview 14
statistics delta 16
Statistics dictionary 15
statistics tuple 16
statistics tuple structure 16
string basic type 10
Structures
base class statistics schema 18
crawl collection statistics schema 21
crawl collections dictionary 15
crawl site statistics schema 20
crawl statistics cycle 16
crawl statistics delta 16
flattened crawl collection statistics 23
flattened global statistics 25
global statistics schema 19
statistics - overview 14
statistics data types 17
Statistics dictionary 15
statistics tuple 16
Syntax - messages 10
T
Timer events
client 37
server 39
Timers
client 37
server 38
timestamp basic type 10
Tracking changes 53
Transport 10
Triggered events - higher-layer
client 37
server 38
Types - basic
double 10
int 10
long 10
overview- 10
string 10
timestamp 10
Types - complex
crawl collection integer status code 13
crawl collection string status code 13
crawl mode string status code 14
log level 13
overview 11
URI skip code 11
web document skip code 12
U
URI skip code simple type 11
V
Vendor-extensible fields 9
Versioning 9
W
Web crawler Management message 26
Web crawler management methods
GetLogLevel 26
GetNodeSchedulerXmlRpcPort 26
GetSiteManagerNum 27
IsDistributed 27
IsMultiNodeScheduler 27
overview 26
SetLogLevel 27
web document skip code simple type 12