Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation for protocols, file formats, languages, standards as well as overviews of the interaction among each of these technologies.

- **Copyrights.** This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you may make copies of it in order to develop implementations of the technologies described in the Open Specifications and may distribute portions of it in your implementations using these technologies or your documentation as necessary to properly document the implementation. You may also distribute in your implementation, with or without modification, any schema, IDL’s, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications.

- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.

- **Patents.** Microsoft has patents that may cover your implementations of the technologies described in the Open Specifications. Neither this notice nor Microsoft’s delivery of the documentation grants any licenses under those or any other Microsoft patents. However, a given Open Specification may be covered by Microsoft Open Specification Promise or the Community Promise. If you would prefer a written license, or if the technologies described in the Open Specifications are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.

- **Trademarks.** The names of companies and products contained in this documentation may be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights.

- **Fictitious Names.** The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

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

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

<table>
<thead>
<tr>
<th>Date</th>
<th>Revision History</th>
<th>Revision Class</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>04/04/2008</td>
<td>0.1</td>
<td></td>
<td>Initial Availability</td>
</tr>
<tr>
<td>04/25/2008</td>
<td>0.2</td>
<td>Major</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>06/27/2008</td>
<td>1.0</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>12/12/2008</td>
<td>1.01</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>03/18/2008</td>
<td>1.02</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>07/13/2009</td>
<td>1.03</td>
<td>Major</td>
<td>Changes made for template compliance</td>
</tr>
<tr>
<td>08/28/2009</td>
<td>1.04</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>11/06/2009</td>
<td>1.05</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>02/19/2010</td>
<td>2.0</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>03/31/2010</td>
<td>2.01</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>04/30/2010</td>
<td>2.02</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>06/07/2010</td>
<td>2.03</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>06/29/2010</td>
<td>2.04</td>
<td>Minor</td>
<td>Clarified the meaning of the technical content.</td>
</tr>
<tr>
<td>07/23/2010</td>
<td>2.04</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>2.04</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>2.04</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>2.05</td>
<td>Editorial</td>
<td>Changed language and formatting in the technical content.</td>
</tr>
<tr>
<td>03/18/2011</td>
<td>2.5.1</td>
<td>Editorial</td>
<td>Changed language and formatting in the technical content.</td>
</tr>
<tr>
<td>06/10/2011</td>
<td>2.5.1</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>2.5.1</td>
<td>No change</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>04/11/2012</td>
<td>2.5.1</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>2.5.1</td>
<td>No change</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
</tbody>
</table>
3.1.4.17 proc_CreateWeb ................................................................. 82
3.1.4.17.1 Audit Flags Result Set .................................................. 82
3.1.4.18 proc_DeleteAllItemVersions .............................................. 85
3.1.4.19 proc_DeleteAttachment .................................................... 86
3.1.4.20 proc_DeleteAttachmentsFolder ......................................... 87
3.1.4.21 proc_DeleteCategory ....................................................... 88
3.1.4.22 proc_DeleteChanges ........................................................ 88
3.1.4.23 proc_DeleteEventLog ....................................................... 88
3.1.4.24 proc_DeleteItemVersion .................................................... 89
3.1.4.25 proc_DeleteSite ............................................................. 90
3.1.4.25.1 Site Collection Flags Result Set ..................................... 90
3.1.4.25.2 Distribution List E-mail Address Result Set ..................... 90
3.1.4.26 proc_DeleteSiteAsync ...................................................... 91
3.1.4.27 proc_GetSiteDeletionBatch ............................................... 91
3.1.4.27.1 Site Collection Deletion Batch Result Set ....................... 91
3.1.4.28 proc_DeleteSiteInternalAsync ........................................... 92
3.1.4.29 proc_DeleteView ............................................................ 92
3.1.4.30 proc_DeleteWeb ............................................................. 93
3.1.4.30.1 Audit Flags Result Set ................................................ 94
3.1.4.31 proc_DropListRecord ...................................................... 95
3.1.4.32 proc_FetchOldDoc .......................................................... 97
3.1.4.32.1 Domain Group Cache Versions Result Set ......................... 98
3.1.4.32.2 Domain Group Cache Back-End Database Server Update Result Set .......................................................... 99
3.1.4.32.3 Domain Group Cache Front-End Web Server Update Result Set .......................................................... 99
3.1.4.32.4 Document Version Metadata Result Set ........................... 99
3.1.4.32.5 Document Version Content Stream Result Set ................ 102
3.1.4.33 proc_FindDocs .............................................................. 103
3.1.4.33.1 Found Docs Result Set ................................................. 104
3.1.4.34 proc_FinishUndirtyList .................................................... 104
3.1.4.35 proc_GenerateUniqueFileName ........................................ 105
3.1.4.35.1 Unique File Name Result Set .......................................... 105
3.1.4.36 proc_GetAllAttachmentsInfo ........................................... 105
3.1.4.36.1 List Attachments Result Set ........................................ 106
3.1.4.36.2 List Item Attachments Result Set .................................. 106
3.1.4.37 proc_GetChanges ........................................................... 107
3.1.4.37.1 EventInformation Result Set ........................................ 107
3.1.4.37.2 EventDetails Result Set .............................................. 108
3.1.4.38 proc_GetCurrent ............................................................ 109
3.1.4.38.1 EventInformation Result Set ........................................ 109
3.1.4.39 proc_GetDocIdUrl .......................................................... 109
3.1.4.40 proc_GetFullLinkInfoForSingleDoc ................................... 110
3.1.4.40.1 Web List For Normalization Result Set ............................ 111
3.1.4.40.2 NULL Individual URL Security Result Set ......................... 111
3.1.4.40.3 Individual URL Security Result Set ................................ 111
3.1.4.40.4 Document Link Information Result Set ........................... 112
3.1.4.40.5 Document Setup Path Result Set ................................... 113
3.1.4.41 proc_GetListDataLinks .................................................... 113
3.1.4.41.1 Web List For Normalization Result Set ............................ 114
3.1.4.41.2 List Data Link Information Result Set ............................ 114
3.1.4.42 proc_GetUrlDocId .......................................................... 117
3.1.4.42.1 Directory And Leaf Names Result Set .............................. 117
3.1.4.43 proc_InsertEventSubscriptionJunctionEntries ....................... 117
3.1.4.44 proc_InsertItemIntoNameValuePair .................................. 131
3.1.4.45 proc_InsertJunction ................................................. 132
3.1.4.46 proc_ListDocsInCategory ............................................ 134
3.1.4.46.1 Docs Category Result Set ....................................... 134
3.1.4.46.2 DocsCategoryMetaInfo Result Set ............................ 135
3.1.4.47 proc_ListThemeFiles ............................................... 135
3.1.4.47.1 Theme Files Information Result Set .......................... 135
3.1.4.48 proc_ListThemes ................................................... 136
3.1.4.48.1 Theme Information Result Set ................................. 137
3.1.4.49 proc_LoadTheme .................................................... 137
3.1.4.49.1 Theme Files Information Result Set .......................... 139
3.1.4.49.2 Theme INF File Info Result Set .............................. 140
3.1.4.50 proc_LogChange .................................................... 140
3.1.4.51 proc_PatchLinkForFile ............................................ 141
3.1.4.52 proc_PatchLinkForWeb ............................................ 142
3.1.4.53 proc_RefreshCheckout ............................................ 143
3.1.4.53.1 Document Metadata Result Set ............................... 144
3.1.4.53.2 NULL Result Set ................................................ 144
3.1.4.54 proc_RemoveJunctions ............................................. 144
3.1.4.55 proc_RenameHostHeaderSite .................................... 145
3.1.4.56 proc_RenameSite .................................................. 146
3.1.4.57 proc_SetNextId .................................................... 146
3.1.4.58 proc_StartUndirtyList ............................................ 147
3.1.4.58.1 Cache Parse Identifier Result Set ............................ 147
3.1.4.59 proc_TakeOfflineDocument .................................... 147
3.1.4.59.1 Document Metadata Result Set ............................... 149
3.1.4.59.2 Event Receivers Result Set .................................. 149
3.1.4.59.3 NULL Result Set ................................................ 149
3.1.4.59.4 Link Info Single Doc Result Set ............................. 149
3.1.4.60 proc_UndirtyListItem ............................................. 149
3.1.4.61 proc_UpdateDirtyDocument ..................................... 150
3.1.4.62 proc_UpdateItemInNameValuePair ............................. 151
3.1.4.63 proc_UpdateOrderNumber ....................................... 153
3.1.4.64 proc_UpdateVersionVirusInfo .................................. 154
3.1.4.65 proc_UpdateView .................................................. 155
3.1.4.66 proc_UpdateVirusInfo ............................................ 157
3.1.4.67 proc_UpdateWebPartLinks ...................................... 158
3.1.4.68 proc_UserHasDataItems ........................................ 159
3.1.4.69 proc_InsertItemIntoNameValuePairCollated .................. 159
3.1.4.70 proc_UpdateItemInTheNameValuePairCollated .............. 160
3.1.5 Timer Events .......................................................... 161
3.1.6 Other Local Events .................................................. 162
3.2 WSSDLIM Client Details ................................................. 162
3.2.1 Abstract Data Model ................................................ 162
3.2.2 Timers ................................................................ 162
3.2.3 Initialization ........................................................... 162
3.2.4 Message Processing Events and Sequencing Rules ............ 163
3.2.5 Timer Events ........................................................... 163
3.2.6 Other Local Events .................................................. 163

4 Protocol Examples ........................................................................ 164
4.1 Associate a Category with a Document .................................. 164
4.2 Remove Association of a Category from a Document .............. 165
4.3 Add a Category to a Site ................................................... 167
1 Introduction

The Windows SharePoint Services: Content Database Document and List Item Management Communications protocol specifies the communication sequences used by the front-end Web server and application servers to perform data query and update commands on back-end database servers as part of document and list item management operations.

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]:

- access control list (ACL)
- access mask
- anonymous user
- application
- Component Object Model (COM)
- Coordinated Universal Time (UTC)
- domain
- file system
- GUID
- Hypertext Transfer Protocol (HTTP)
- language code identifier (LCID)
- security policy
- Unicode
- XML

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

- 12-hour clock notation
- 24-hour clock notation
- action
- alert
- alert subscription
- assembly
- assembly name
- attachment
- audit flag
- author
- back-end database server
- bot
- calendar type
- CAML
- cascading style sheet (CSS)
- category
- change log
- change log identifier
- check in
- check out
- checked out
- Collaborative Application Markup Language (CAML)
home page
host header
HTTP entity tag
HTTP GET
HTTP HEAD
hyperlink
indexed field
internal version number
Internet Information Services (IIS)
item
item identifier
leaf name
link fixup
list
list identifier
list item
list item identifier
list template
list view
locked
login name
lookup field
major version
master page
Meeting Workspace site
member
metadict
minor version
moderated object
page navigation dependency
page type
parent site
path segment
permission
permission level
personal view
policy
provisioned
publish
published
published version
publishing level
query
read-only mode
Recycle Bin
result set
return code
role
role assignment
role definition
row
row ordinal
schema version
scope identifier
security group
security principal
security provider
security scope
server-relative URL
shared view
Simple Object Access Protocol (SOAP)
site
site collection
site collection administrator
site collection flag
site collection identifier
site collection quota
site definition
site definition version
site identifier
site-relative URL
static page
stored procedure
store-relative form
subsite
SystemID
theme
thicket
thicket folder
thicket supporting file
top-level site
transaction
Transact-Structured Query Language (T-SQL)
uncustomized
Uniform Resource Locator (URL)
user identifier
user interface (UI) version
version
view
view flag
view identifier
virus scanner
Web application
Web bot
Web Part
Web Part identifier
Web Part Page
Web Part property
Web Part type identifier
Web Part zone
Welcome page
wide list
workflow
workflow identifier
XML namespace
XML schema definition (XSD)
zero-based index

The following terms are specific to this document:

[MS-WSSDLIM] — v20120630
Windows SharePoint Services: Content Database Document and List Item Management Communications Protocol Specification

Copyright © 2012 Microsoft Corporation.

Release: July 16, 2012
**calculated field:** A user-defined field that can perform calculations by using the contents of other fields.

**list flag:** An 8-byte unsigned integer bit mask that provides metadata about a SharePoint list.

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

### 1.2 References

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

#### 1.2.1 Normative References

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


[MS-WSSCCSP] Microsoft Corporation, "Windows SharePoint Services: Content Database Core List Schema and Site Provisioning Communications Protocol Specification".


1.2.2 Informative References


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

1.3 Protocol Overview (Synopsis)

This protocol specifies the communication between the front-end Web server and the back-end database server used to satisfy requests involving the following operations:

1.3.1 Category Operations

Includes methods to associate strings, called categories (2), with documents categories are retrievable using the site metadata key "vti_categories." Categories have meanings defined by applications, and applications can implement dynamic behavior for sites or documents that have a particular category associated with them.

1.3.2 Change Log Operations

Includes methods to retrieve or append entries to the change log. The Change Log contains information about item actions such as Add, Update, Delete, Rename, Move Away, and Move Into, at the list, Site, site collection, and content database levels. An application could use this information to find out what changes have occurred on the data objects stored in the back-end database server. It could further use this information to implement synchronization features by replaying these events on a different Site.

1.3.3 Publish and Un-publish Operations

Includes methods to change the publishing level of a document in a list.

1.3.4 Check In and Check Out Operations

Includes methods to check out and check in a document in a list.

1.3.5 Historical Versioning Operations

Includes methods for managing the historical versions of a document or list item.

1.3.6 Link Fixup Operations

A back-end database server and front-end Web server work together to implement link fixup. The back-end database server tracks forward links from list items to documents. When the back-end database server performs an operation that requires link fixup for list items, it defers potentially complex work by marking the list items needing link fixup as dirty. Later, before a Front-End Web Server retrieves list data, it checks the "list is dirty" status and, if the list is dirty, performs a link fixup operation before querying the data in the list.
1.3.7 **Theme Operations**

Includes methods to retrieve theme information for the purposes of applying a theme to the pages that belong to a site.

1.3.8 **Wide List Operations**

Includes methods to create an unlimited number of fields for lists. The Content Database for a back-end database server has a fixed schema. To accommodate for this, the concept of wide list and row ordinal were created.

If the number of fields in a list of a particular type exceeds a fixed maximum for that type per Row in the Content Database, then new rows are allocated for every list item in the list.

1.4 **Relationship to Other Protocols**

The following diagram shows the transport stack that the protocol uses:

![Transport Stack Diagram](image)

Figure 1: This protocol in relation to other protocols

1.5 **Prerequisites/Preconditions**

The operations described by the protocol operate between a client and a back end-database server on which the databases are stored. The client is expected to know the location and connection information for the databases.

This protocol requires that the protocol client has appropriate permissions to call the stored procedures stored on the back-end database server.

1.6 **Applicability Statement**

This protocol is intended for use by protocol clients and protocol servers that are both connected by high-bandwidth, low latency network connections.

1.7 **Versioning and Capability Negotiation**

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

1.8 **Vendor-Extensible Fields**

None.
1.9 Standards Assignments

None.
2 Messages

2.1 Transport

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

2.2 Common Data Types

This section contains common definitions used by this protocol.

2.2.1 Simple Data Types and Enumerations

2.2.1.1 Change Log ListId

Change Log ListId is a GUID and is part of a Change Log Entry. This data has different meanings based on the Event Object Type and Event Type data of the same Change Log Entry. The different meanings of this value are specified as the following:

Event Object Type = 0x00000001 (List Item)

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00000100</td>
<td>The identifier of the list that contains the list item.</td>
</tr>
<tr>
<td>0x00000200</td>
<td>The identifier of the list that contains the list item.</td>
</tr>
<tr>
<td>0x00000400</td>
<td>The identifier of the list that contains the list item.</td>
</tr>
<tr>
<td>0x00000800</td>
<td>The identifier of the list that contains the list item.</td>
</tr>
<tr>
<td>0x00010000</td>
<td>The identifier of the list to which this list item was moved.</td>
</tr>
<tr>
<td>0x00020009</td>
<td>The identifier of the list that contains the list item.</td>
</tr>
<tr>
<td>or 0x00020000</td>
<td></td>
</tr>
<tr>
<td>0x00080000</td>
<td>The identifier of the list that contains the list item.</td>
</tr>
<tr>
<td>0x01000000</td>
<td>The identifier of the list that contains the list item.</td>
</tr>
<tr>
<td>0x02000000</td>
<td>The identifier of the list that contains the list item.</td>
</tr>
<tr>
<td>0x04000000</td>
<td>The identifier of the list that contains the list item.</td>
</tr>
</tbody>
</table>

Event Object Type = 0x00000002 (List)
<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00001000</td>
<td>The identifier of the list.</td>
</tr>
<tr>
<td>0x00002000</td>
<td>The identifier of the list.</td>
</tr>
<tr>
<td>0x00004000</td>
<td>The identifier of the list.</td>
</tr>
<tr>
<td>0x00008000</td>
<td>The identifier of the list.</td>
</tr>
<tr>
<td>0x00020000</td>
<td>The identifier of the list.</td>
</tr>
<tr>
<td>0x00080000</td>
<td>The identifier of the list.</td>
</tr>
<tr>
<td>0x02000000</td>
<td>The identifier of the list.</td>
</tr>
</tbody>
</table>

Event Object Type = 0x00000004 (Site)

MUST be NULL.

Event Object Type = 0x00000008 (Site Collection)

MUST be NULL.

Event Object Type = 0x00000010 (File)

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00001000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00004000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00010000</td>
<td>The identifier of the list into which the file is being moved.</td>
</tr>
<tr>
<td>0x00020000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00008000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00100000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x04000000</td>
<td>The identifier of the list from which the file is being moved.</td>
</tr>
</tbody>
</table>

Event Object Type = 0x00000020 (Folder)

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00001000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00002000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>Event Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-------------</td>
</tr>
<tr>
<td>0x00004000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00008000</td>
<td>The identifier of the list into which the folder is moved.</td>
</tr>
<tr>
<td>0x00010000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00020000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00100000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x04000000</td>
<td>The identifier of the list which the folder used to belong.</td>
</tr>
</tbody>
</table>

Event Object Type = 0x00000040 (Alert Subscription)

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00001000</td>
<td>The identifier of the list that contains this alert subscription</td>
</tr>
<tr>
<td>0x00002000</td>
<td>The identifier of the list that contains this alert subscription.</td>
</tr>
<tr>
<td>0x00004000</td>
<td>The identifier of the list that contains this alert subscription.</td>
</tr>
</tbody>
</table>

Event Object Type = 0x00000080 (Security Principal)
MUST be NULL.

Event Object Type = 0x00000100 (Security Group)
MUST be NULL.

Event Object Type = 0x00000200 (Content Type)
MUST be NULL.

Event Object Type = 0x00000400 (Field Template)
MUST be NULL.

Event Object Type = 0x00000800 (Security Policy)
MUST be NULL.

Event Object Type = 0x00001000 (View)
<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00001000</td>
<td>The identifier of the list that contains the view.</td>
</tr>
<tr>
<td>0x00002000</td>
<td>The identifier of the list that contains the view.</td>
</tr>
<tr>
<td>0x00004000</td>
<td>The identifier of the list that contains the view.</td>
</tr>
</tbody>
</table>

### 2.2.1.2 Change Log ItemId

Change Log **ItemId** is an integer and is part of a Change Log Entry. This data has different meanings based on the Event Object Type and Event Type data of the same Change Log Entry. The different meanings of this value are specified as the following:

**Event Object Type = 0x00000001 (List Item)**

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00000010</td>
<td>The <strong>item identifier</strong> of the list item.</td>
</tr>
<tr>
<td>0x00000020</td>
<td>The item identifier of the list item.</td>
</tr>
<tr>
<td>0x00000040</td>
<td>The item identifier of the list item.</td>
</tr>
<tr>
<td>0x00000080</td>
<td>The item identifier of the list item.</td>
</tr>
<tr>
<td>0x00000100</td>
<td>The item identifier of the list item.</td>
</tr>
<tr>
<td>0x00001001</td>
<td>The item identifier of the list item.</td>
</tr>
<tr>
<td>0x00002000</td>
<td>The item identifier of the list item.</td>
</tr>
<tr>
<td>0x00002002</td>
<td>The item identifier of the list item.</td>
</tr>
<tr>
<td>0x00004004</td>
<td>The item identifier of the list item.</td>
</tr>
<tr>
<td>0x00008000</td>
<td>The item identifier of the list item.</td>
</tr>
<tr>
<td>0x00100000</td>
<td>The item identifier of the list item.</td>
</tr>
<tr>
<td>0x02000000</td>
<td>The item identifier of the list item.</td>
</tr>
<tr>
<td>0x04000000</td>
<td>The old item identifier of the list item.</td>
</tr>
</tbody>
</table>

**Event Object Type = 0x00000002 (List)**

**MUST be NULL.**

**Event Object Type = 0x00000004 (Site)**

**MUST be NULL.**
Event Object Type = 0x00000008 (Site Collection)
MUST be NULL.

Event Object Type = 0x00000010 (File)

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00001000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00004000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00010000</td>
<td>The item identifier of the file inside the list which it is being moved into.</td>
</tr>
<tr>
<td>0x00020000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00080000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x01000000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x04000000</td>
<td>The item identifier of the file inside the list which it is being moved away from.</td>
</tr>
</tbody>
</table>

Event Object Type = 0x00000020 (Folder)

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00001000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00002000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00004000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00080000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x01000000</td>
<td>The item identifier of the folder in the list after the move.</td>
</tr>
<tr>
<td>0x02000000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x01000000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x04000000</td>
<td>The item identifier of the folder in the list before the move.</td>
</tr>
</tbody>
</table>

Event Object Type = 0x00000040 (Alert Subscription)

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00001000</td>
<td>The item identifier of the document, if this alert subscription is associated with this list item. NULL, if the alert subscription is associated with the list.</td>
</tr>
<tr>
<td>0x00020000</td>
<td>The item identifier of the document.</td>
</tr>
<tr>
<td>0x00040000</td>
<td>The item identifier of the document.</td>
</tr>
</tbody>
</table>

Event Object Type = 0x00000080 (Security Principal)

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00001000</td>
<td>Equal to Int0 if the security principal added is active; otherwise MUST be NULL.</td>
</tr>
</tbody>
</table>
### Event Type Description

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00002000</td>
<td>SHOULD be the Security Principal Identifier, if the Security Principal is active after the update, otherwise, NULL&lt;sup&gt;1&lt;/sup&gt;</td>
</tr>
<tr>
<td>0x00004000</td>
<td>MUST be NULL.</td>
</tr>
</tbody>
</table>

Event Object Type = 0x00000100 (Security Group)

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00001000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00002000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00004000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00200000</td>
<td>The identifier of the security principal which is added to the security group.</td>
</tr>
<tr>
<td>0x00400000</td>
<td>The identifier of the security principal which is removed from the security group.</td>
</tr>
</tbody>
</table>

Event Object Type = 0x00000200 (Content Type)

MUST be NULL.

Event Object Type = 0x00000400 (Field Template)

MUST be NULL.

Event Object Type = 0x00000800 (Security Policy)

MUST be NULL.

Event Object Type = 0x00001000 (View)

MUST be NULL.

### 2.2.1.3 Change Log DocId

Change Log DocId is a GUID and is part of a Change Log Entry. This data has different meanings based on the Event Object Type and Event Type data of the same Change Log Entry. The different meanings of this value are specified as the following:

Event Object Type = 0x00000001 (List Item)

The **document identifier** of the document associated with this list item.

Event Object Type = 0x00000002 (List)

MUST be NULL.

Event Object Type = 0x00000004 (Site)

MUST be NULL.

Event Object Type = 0x00000008 (Site Collection)

MUST be NULL.
MUST be NULL.
Event Object Type = 0x00000010 (File)
The document identifier of the document.
Event Object Type = 0x00000020 (Folder)
The identifier of the folder.
Event Object Type = 0x00000040 (Alert Subscription)
MUST be NULL.
Event Object Type = 0x00000080 (Security Principal)
MUST be NULL.
Event Object Type = 0x00000100 (Security Group)
MUST be NULL.
Event Object Type = 0x00000200 (Content Type)
MUST be NULL.
Event Object Type = 0x00000400 (Field Template)
MUST be NULL.
Event Object Type = 0x00000800 (Security Policy)
MUST be NULL.
Event Object Type = 0x00001000 (View)

2.2.1.4 Change Log Guid0

Change Log Guid0 is a GUID and is part of a Change Log Entry. This data has different meanings based on the Event Object Type and Event Type data of the same Change Log Entry. The different meanings of this value are specified as the following:

Event Object Type = 0x00000001 (List Item)

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00000010</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00000020</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00000040</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00000080</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00000100</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00001001</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00002000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>Event Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>0x00002002</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00004004</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00008000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00010000</td>
<td>The identity of the list used to contain the list item.</td>
</tr>
<tr>
<td>0x00020009</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00080000</td>
<td>Id of the Scope where the role assignments are added.</td>
</tr>
<tr>
<td>0x00100000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x02000000</td>
<td>Id of the Scope where the Role Assignments are deleted.</td>
</tr>
<tr>
<td>0x04000000</td>
<td>The identifier of the list that this list item was moved to.</td>
</tr>
</tbody>
</table>

Event Object Type = 0x00000002 (List)

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00001000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00002000</td>
<td>When a Field is deleted from the List, this value is the identifier of the Field. Otherwise, this value MUST be NULL.</td>
</tr>
<tr>
<td>0x00004000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00008000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00020000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00080000</td>
<td>Id of the Scope where the Role Assignment is added.</td>
</tr>
<tr>
<td>0x00080000</td>
<td>Id of the Scope where the Role Assignments are added.</td>
</tr>
<tr>
<td>0x02000000</td>
<td>Id of the Scope where the Role Assignment is deleted.</td>
</tr>
<tr>
<td>0x02000000</td>
<td>Id of the Scope where the Role Assignments are deleted.</td>
</tr>
</tbody>
</table>

Event Object Type = 0x00000004 (Site)

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00001000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00002000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00004000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00008000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00040000</td>
<td>Id of the Scope where the role definition is added.</td>
</tr>
<tr>
<td>0x00020000</td>
<td>MUST be NULL.</td>
</tr>
</tbody>
</table>
### Event Type

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00800000</td>
<td>Id of the Scope where the role definition is deleted.</td>
</tr>
<tr>
<td>0x01000000</td>
<td>Id of the Scope where the role definition is modified.</td>
</tr>
<tr>
<td>0x02000000</td>
<td>Id of the Scope where the Role Assignment is deleted.</td>
</tr>
<tr>
<td>0x00080000</td>
<td>Id of the Scope where the Role Assignment is added.</td>
</tr>
<tr>
<td>0x08000000</td>
<td>MUST be NULL.</td>
</tr>
</tbody>
</table>

Event Object Type = 0x00000008 (Site Collection)

MUST be NULL.

Event Object Type = 0x00000010 (File)

MUST be NULL.

Event Object Type = 0x00000020 (Folder)

MUST be NULL.

Event Object Type = 0x00000040 (Alert Subscription)

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00001000</td>
<td>The identifier of the alert subscription.</td>
</tr>
<tr>
<td>0x00002000</td>
<td>The identifier of the alert subscription.</td>
</tr>
<tr>
<td>0x00004000</td>
<td>The identifier of the alert subscription.</td>
</tr>
</tbody>
</table>

Event Object Type = 0x00000080 (Security Principal)

MUST be NULL.

Event Object Type = 0x00000100 (Security Group)

MUST be NULL.

Event Object Type = 0x00000200 (Content Type)

MUST be NULL.

Event Object Type = 0x00000400 (Field Template)

MUST be NULL.

Event Object Type = 0x00000800 (Security Policy)

MUST be NULL.
Event Object Type = 0x00001000 (View)

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00001000</td>
<td>The identifier of the list view.</td>
</tr>
<tr>
<td>0x00002000</td>
<td>The identifier of the alert subscription.</td>
</tr>
<tr>
<td>0x00004000</td>
<td>The identifier of the alert subscription.</td>
</tr>
</tbody>
</table>

2.2.1.5 Change Log Int0

Change Log Int0 is an integer and is part of a Change Log Entry. This data has different meanings based on the Event Object Type and Event Type data of the same Change Log Entry. The different meanings of this value are specified as the following:

Event Object Type = 0x00000001 (List Item)

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00000010</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00000020</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00000040</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00000080</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00000100</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00001001</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00002000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00002002</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00004004</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00008000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00100000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x02000000</td>
<td>If the Change Log Int1 of the same Change Log Entry (&quot;Int1&quot;) is NULL, this value MUST be NULL. If Int1 is greater than or equal to 0x40000000, this value is the Security Principal Identifier of the role assignment that is added. If Int1 is NOT NULL and less than 0x40000000, this value is the Security Principal Identifier which has made the role inheritance change.</td>
</tr>
<tr>
<td>0x04000000</td>
<td>Current Identifier of the List Item.</td>
</tr>
</tbody>
</table>
### Event Object Type = 0x0000002 (List)

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00001000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00002000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00004000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00008000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00020000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00080000</td>
<td>If the Change Log Int1 of the same Change Log Entry (&quot;Int1&quot;) is NULL, this value MUST be -1. If Int1 is greater than or equal to 0x40000000, this value is the Security Principal Identifier of the role assignment that is added. If Int1 is NOT NULL and less than 0x40000000, this value is the Security Principal Identifier which has made the role inheritance change.</td>
</tr>
<tr>
<td>0x02000000</td>
<td>If the Change Log Int1 of the same Change Log Entry (&quot;Int1&quot;) is NULL, this value is the Security Principal Identifier that is removed from all Roles. If Int1 is greater than or equal to 0x40000000, this value is the Security Principal Identifier that is removed from the Role specified by Int1. If Int1 is not NULL, and less than 0x40000000, this value MUST be NULL.</td>
</tr>
</tbody>
</table>

### Event Object Type = 0x0000004 (Site)

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00001000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00002000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00004000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00008000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00040000</td>
<td>SHOULD be the Security Principal Identifier who is adding this role definition.</td>
</tr>
<tr>
<td>0x00020000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00800000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x01000000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x02000000</td>
<td>If the Change Log Int1 of the same Change Log Entry (&quot;Int1&quot;) is NULL, this value is the Security Principal Identifier that is removed from all Roles. If Int1 is greater than or equal to 0x40000000, this value is the Security Principal Identifier that is removed from the Role specified by Int1. If Int1 is not NULL, and less than 0x40000000, this value MUST be NULL.</td>
</tr>
<tr>
<td>0x00080000</td>
<td>If the Change Log Int1 of the same Change Log Entry (&quot;Int1&quot;) is NULL, this value MUST be -1. If Int1 is greater than or equal to 0x40000000, this value is the Security Principal Identifier of the role assignment that is added. If Int1 is NOT NULL and less than 0x40000000, this value is the Security Principal Identifier which has made the role inheritance change.</td>
</tr>
<tr>
<td>0x08000000</td>
<td>MUST be NULL.</td>
</tr>
</tbody>
</table>

### Event Object Type = 0x0000008 (Site Collection)
MUST be NULL.
Event Object Type = 0x00000010 (File)
MUST be NULL.
Event Object Type = 0x00000020 (Folder)
MUST be NULL.
Event Object Type = 0x00000040 (Alert Subscription)
MUST be NULL.
Event Object Type = 0x00000080 (Security Principal)

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00001000</td>
<td>Id of the Security Principal added</td>
</tr>
<tr>
<td>0x00002000</td>
<td>User Identifier</td>
</tr>
<tr>
<td>0x00004000</td>
<td>Security Principal Identifier</td>
</tr>
</tbody>
</table>

Event Object Type = 0x00000100 (Security Group)
MUST be Group Id.

Event Object Type = 0x00000200 (Content Type)
MUST be NULL.

Event Object Type = 0x00000400 (Field Template)
MUST be NULL.

Event Object Type = 0x00000800 (Security Policy)
MUST be NULL.

Event Object Type = 0x00001000 (View)
MUST be NULL.

2.2.1.6 Change Log ContentTypeId

Change Log ContentTypeId is a numeric string value of arbitrary but limited length. It is part of a Change Log Entry. This data has different meanings based on the Event Object Type and Event Type data of the same Change Log Entry. The different meanings of this value are specified as the following:

Event Object Type = 0x00000010 (List Item)

MUST be NULL.
Event Object Type = 0x00000002 (List)
MUST be NULL.

Event Object Type = 0x00000004 (Site)
MUST be NULL.

Event Object Type = 0x00000008 (Site Collection)
MUST be NULL.

Event Object Type = 0x00000010 (File)
MUST be NULL.

Event Object Type = 0x00000020 (Folder)
MUST be NULL.

Event Object Type = 0x00000040 (Alert Subscription)
MUST be NULL.

Event Object Type = 0x00000080 (Security Principal)
MUST be NULL.

Event Object Type = 0x00000100 (Security Group)
MUST be NULL.

Event Object Type = 0x00000200 (Content Type)

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00001000</td>
<td>The content type identifier of the content type.</td>
</tr>
<tr>
<td>0x00002000</td>
<td>The content type identifier of the content type.</td>
</tr>
<tr>
<td>0x00004000</td>
<td>The content type identifier of the content type.</td>
</tr>
</tbody>
</table>

Event Object Type = 0x00000400 (Field Template)

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00000400</td>
<td>The identifier of the field.</td>
</tr>
</tbody>
</table>

Event Object Type = 0x00000800 (Security Policy)
MUST be NULL.

Event Object Type = 0x00001000 (View)
MUST be NULL.
### 2.2.1.7 Change Log ItemFullUrl

Change Log ItemFullUrl is a Uniform Resource Locator (URL) of store-relative form. It is part of a Change Log Entry. This data has different meanings based on the Event Object Type and Event Type data of the same Change Log Entry. The different meanings of this value are specified as the following:

**Event Object Type = 0x00000001 (List Item)**

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00000010</td>
<td>URL in Store-relative form of the document.</td>
</tr>
<tr>
<td>0x00000020</td>
<td>URL in Store-relative form of the document.</td>
</tr>
<tr>
<td>0x00000040</td>
<td>URL in Store-relative form of the document.</td>
</tr>
<tr>
<td>0x00000080</td>
<td>URL in Store-relative form of the document.</td>
</tr>
<tr>
<td>0x00000100</td>
<td>URL in Store-relative form of the document.</td>
</tr>
<tr>
<td>0x00000101</td>
<td>URL in Store-relative form of the document.</td>
</tr>
<tr>
<td>0x00000200</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00002002</td>
<td>URL in Store-relative form of the document.</td>
</tr>
<tr>
<td>0x00004004</td>
<td>URL in Store-relative form of the document.</td>
</tr>
<tr>
<td>0x00008000</td>
<td>The new URL in Store-relative form of the document.</td>
</tr>
<tr>
<td>0x00010000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00020009</td>
<td>URL in Store-relative form of the document.</td>
</tr>
<tr>
<td>0x00080000</td>
<td>URL of the Scope where the Role Assignment is added. This value MUST be NULL if the Change Log Int0 of the same Change Log Entry is also NULL.</td>
</tr>
<tr>
<td>0x00100000</td>
<td>URL in Store-relative form of the document.</td>
</tr>
<tr>
<td>0x02000000</td>
<td>URL in Store-relative Form to the security scope.</td>
</tr>
<tr>
<td>0x04000000</td>
<td>MUST be NULL.</td>
</tr>
</tbody>
</table>

**Event Object Type = 0x00000002 (List)**

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00010000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00020000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00080000</td>
<td>URL in Store-relative form of the List.</td>
</tr>
<tr>
<td>0x00008000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00020000</td>
<td>MUST be NULL.</td>
</tr>
</tbody>
</table>
| 0x00080000 | URL of the Scope where the Role Assignment is added. This value MUST be NULL if Change
Event Type | Description
---|---
Log Int0 of the same Change Log Entry is NULL.
0x02000000 | URL in Store-relative Form to the **security scope**.

Event Object Type = 0x00000004 (Site)

Event Type | Description
---|---
0x00001000 | MUST be NULL.
0x00002000 | MUST be NULL.
0x00004000 | URL in Store-relative form of the site, if this Site is deleted. This value MUST be NULL if the Site is converted into a folder under its **parent site**.
0x00008000 | MUST be NULL.
0x00040000 | URL in the Store-relative Form of this Site.
0x00020000 | MUST be NULL.
0x00800000 | URL in Store-relative Form to the security scope.
0x01000000 | URL in Store-relative Form to the security scope.
0x02000000 | URL in Store-relative Form to the security scope.
0x00080000 | URL in Store-relative Form to the security scope.
0x08000000 | MUST be NULL.

Event Object Type = 0x00000008 (Site Collection)

MUST be NULL.

Event Object Type = 0x00000010 (File)

SHOULD be NON-NULL. If NOT NULL it MUST be store-relative form of the document.

Event Object Type = 0x00000020 (Folder)

Event Type | Description
---|---
0x00001000 | URL in Store-relative form of the Folder.
0x00002000 | URL in Store-relative form of the Folder.
0x00004000 | URL in Store-relative form of the Folder.
0x00008000 | New URL in Store-relative form of the Folder.
0x00100000 | New URL in Store-relative form of the Folder after the move.
0x00020000 | URL in Store-relative form of the Folder.
<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00100000</td>
<td>URL in Store-relative form of the Folder.</td>
</tr>
<tr>
<td>0x04000000</td>
<td>URL in Store-relative form of the Folder after the move.</td>
</tr>
</tbody>
</table>

Event Object Type = 0x00000040 (Alert Subscription)
MUST be NULL.

Event Object Type = 0x00000080 (Security Principal)
MUST be NULL.

Event Object Type = 0x00000100 (Security Group)
MUST be NULL.

Event Object Type = 0x00000200 (Content Type)
MUST be NULL.

Event Object Type = 0x00000400 (Field Template)
MUST be NULL.

Event Object Type = 0x00000800 (Security Policy)
MUST be NULL.

Event Object Type = 0x00001000 (View)
MUST be NULL.

2.2.1.8 Change Log TimeLastModified

Change Log TimeLastModified is a timestamp and is part of a Change Log Entry. This data has different meanings based on the Event Object Type and Event Type data of the same Change Log Entry. The different meanings of this value are specified as the following:

Event Object Type = 0x00000001 (List Item)

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00000010</td>
<td>Time when the comment is created.</td>
</tr>
<tr>
<td>0x00000020</td>
<td>Time when the comment is updated.</td>
</tr>
<tr>
<td>0x00000040</td>
<td>Time when the comment is deleted.</td>
</tr>
<tr>
<td>0x00000080</td>
<td>Time when the comment is closed.</td>
</tr>
<tr>
<td>0x00000100</td>
<td>Time when the comment is activated.</td>
</tr>
<tr>
<td>0x00001000</td>
<td>Time when the Add happened.</td>
</tr>
<tr>
<td>0x00002000</td>
<td>Time when the event happened.</td>
</tr>
<tr>
<td>Event Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>0x00002002</td>
<td>Time when the List Item is updated.</td>
</tr>
<tr>
<td>0x00004004</td>
<td>Time when the delete happened.</td>
</tr>
<tr>
<td>0x00008000</td>
<td>Time when the rename happened.</td>
</tr>
<tr>
<td>0x00010000</td>
<td>Time when the restore happened.</td>
</tr>
<tr>
<td>0x00020009</td>
<td>Time when the event happened.</td>
</tr>
<tr>
<td>0x00080000</td>
<td>Time when this update happened.</td>
</tr>
<tr>
<td>0x00100000</td>
<td>Time when the update happened.</td>
</tr>
<tr>
<td>0x02000000</td>
<td>Time when the event happened.</td>
</tr>
<tr>
<td>0x04000000</td>
<td>Time when the move happened.</td>
</tr>
</tbody>
</table>

Event Object Type = 0x00000002 (List)

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00001000</td>
<td>Time when the list was created.</td>
</tr>
<tr>
<td>0x00002000</td>
<td>Time when the update happened.</td>
</tr>
<tr>
<td>0x00004000</td>
<td>Time when the list was deleted.</td>
</tr>
<tr>
<td>0x00008000</td>
<td>Time when the list was renamed.</td>
</tr>
<tr>
<td>0x00020000</td>
<td>Time of occurrence when the list was restored from the Recycle Bin.</td>
</tr>
<tr>
<td>0x00080000</td>
<td>Time when the event happened.</td>
</tr>
<tr>
<td>0x02000000</td>
<td>Time when the event happened.</td>
</tr>
</tbody>
</table>

Event Object Type = 0x00000004 (Site)

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00001000</td>
<td>Time when the site was created.</td>
</tr>
<tr>
<td>0x00002000</td>
<td>Time of the update.</td>
</tr>
<tr>
<td>0x00004000</td>
<td>Time when the event happened.</td>
</tr>
<tr>
<td>0x00008000</td>
<td>Time when the site was renamed.</td>
</tr>
<tr>
<td>0x00040000</td>
<td>Time when the event happened.</td>
</tr>
<tr>
<td>0x00020000</td>
<td>Time when the event happened.</td>
</tr>
<tr>
<td>0x00800000</td>
<td>Time when the event happened.</td>
</tr>
<tr>
<td>0x01000000</td>
<td>Time when the event happened.</td>
</tr>
<tr>
<td>0x02000000</td>
<td>Time when the event happened.</td>
</tr>
<tr>
<td>Event Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>0x00080000</td>
<td>Time when the event happened.</td>
</tr>
<tr>
<td>0x08000000</td>
<td>Time when the site navigation was updated.</td>
</tr>
</tbody>
</table>

Event Object Type = 0x00000008 (Site Collection)

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00001000</td>
<td>Time when the creation happened</td>
</tr>
<tr>
<td>0x00002000</td>
<td>Time when the event happened.</td>
</tr>
<tr>
<td>0x00004000</td>
<td>Time when the event happened.</td>
</tr>
<tr>
<td>0x00008000</td>
<td>Time when the event happened.</td>
</tr>
</tbody>
</table>

Event Object Type = 0x00000010 (File)

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00001000</td>
<td>Time when the addition happened.</td>
</tr>
<tr>
<td>0x00004000</td>
<td>Time when the deletion happened.</td>
</tr>
<tr>
<td>0x00010000</td>
<td>Time when the move happened.</td>
</tr>
<tr>
<td>0x00020000</td>
<td>Time when the restore occurred.</td>
</tr>
<tr>
<td>0x00008000</td>
<td>Time when the rename happens.</td>
</tr>
<tr>
<td>0x00100000</td>
<td>Time when the update happened.</td>
</tr>
<tr>
<td>0x04000000</td>
<td>Time when the move happened.</td>
</tr>
</tbody>
</table>

Event Object Type = 0x00000020 (Folder)

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00001000</td>
<td>Time when the restore happened.</td>
</tr>
<tr>
<td>0x00002000</td>
<td>Time when the update happened.</td>
</tr>
<tr>
<td>0x00004000</td>
<td>Time when the delete happened.</td>
</tr>
<tr>
<td>0x00008000</td>
<td>Time when the rename happened.</td>
</tr>
<tr>
<td>0x00010000</td>
<td>Time when the folder was moved into a list.</td>
</tr>
<tr>
<td>0x00020000</td>
<td>Time when the rename happened.</td>
</tr>
<tr>
<td>0x00100000</td>
<td>Time when the folder is updated.</td>
</tr>
<tr>
<td>0x04000000</td>
<td>Time when the rename happened.</td>
</tr>
</tbody>
</table>

Event Object Type = 0x00000040 (Alert Subscription)
Time when the event happened.
Event Object Type = 0x00000080 (Security Principal)
Time when the event happened.
Event Object Type = 0x00000100 (Security Group)
Time when the event happened.
Event Object Type = 0x00000200 (Content Type)

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00001000</td>
<td>Time when the content type add happened.</td>
</tr>
<tr>
<td>0x00002000</td>
<td>Time when the content type update happened.</td>
</tr>
<tr>
<td>0x00004000</td>
<td>Time when the content type deletion happened.</td>
</tr>
</tbody>
</table>

Event Object Type = 0x00000400 (Field Template)
Time when the event happened.
Event Object Type = 0x00000800 (Security Policy)
Time when the event happened.
Event Object Type = 0x00001000 (View)
Time when the event happened.

2.2.1.9 Change Log ItemName

Change Log ItemName is a string and is part of a Change Log Entry. This data has different meanings based on the Event Object Type and Event Type data of the same Change Log Entry. The different meanings of this value are specified as the following:

Event Object Type = 0x00000001 (List Item)

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00000010</td>
<td>Leaf name of the document.</td>
</tr>
<tr>
<td>0x00000020</td>
<td>Leaf name of the document.</td>
</tr>
<tr>
<td>0x00000040</td>
<td>Leaf name of the document.</td>
</tr>
<tr>
<td>0x00000080</td>
<td>Leaf name of the document.</td>
</tr>
<tr>
<td>0x00000100</td>
<td>Leaf name of the document.</td>
</tr>
<tr>
<td>0x00001001</td>
<td>Leaf name of the document.</td>
</tr>
<tr>
<td>0x00002000</td>
<td>SHOULD be NULL. &lt;3&gt;</td>
</tr>
<tr>
<td>0x00002002</td>
<td>Leaf name of the document.</td>
</tr>
<tr>
<td>Event Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>------------------------------------------------------------------</td>
</tr>
<tr>
<td>0x00004004</td>
<td>Name of the object.</td>
</tr>
<tr>
<td>0x00008000</td>
<td>The Leaf Name of the List Item before it is renamed.</td>
</tr>
<tr>
<td>0x00010000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00020009</td>
<td>List Item name.</td>
</tr>
<tr>
<td>0x00080000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00100000</td>
<td>List Item name OR NULL.</td>
</tr>
<tr>
<td>0x02000000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x04000000</td>
<td>MUST be NULL.</td>
</tr>
</tbody>
</table>

Event Object Type = 0x00000002 (List)

MUST be NULL.

Event Object Type = 0x00000004 (Site)

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00010000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00020000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00040000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00080000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x01000000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x02000000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00080000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x08000000</td>
<td>MUST be NULL.</td>
</tr>
</tbody>
</table>

Event Object Type = 0x00000008 (Site Collection)

MUST be NULL.

Event Object Type = 0x00000010 (File)

SHOULD be NULL,<4>

Event Object Type = 0x00000020 (Folder)
MUST be NULL.
Event Object Type = 0x00000040 (Alert Subscription)
MUST be NULL.
Event Object Type = 0x00000080 (Security Principal)
MUST be NULL.
Event Object Type = 0x00000100 (Security Group)
MUST be NULL except for the following event type:

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00004000</td>
<td>Group Title</td>
</tr>
</tbody>
</table>

Event Object Type = 0x00000200 (Content Type)
MUST be NULL.
Event Object Type = 0x00000400 (Field Template)
MUST be NULL.
Event Object Type = 0x00000800 (Security Policy)
MUST be NULL.
Event Object Type = 0x00001000 (View)
MUST be NULL.

### 2.2.1.10 Change Log Int1

Change Log Int1 is an integer and is part of a Change Log Entry. This data has different meanings based on the Event Object Type and Event Type data of the same Change Log Entry. The different meanings of this value are specified as the following:

Event Object Type = 0x00000001 (List Item)

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00000010</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00000020</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00000040</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00000080</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00000100</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00001001</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>Event Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>0x00002000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00002002</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00004004</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00008000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00010000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00020009</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00080000</td>
<td>If the value is NULL, it indicates that the <strong>anonymous user permission</strong> is updated. If this value is greater than or equal to 0x40000000, it indicates that a role assignment is added. If the value is NOT NULL and less than 0x40000000, it is a security change Type Flag, and indicates that a role inheritance has changed.</td>
</tr>
<tr>
<td>0x00100000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x02000000</td>
<td>If this value is NOT NULL and greater than or equal to 0x40000000, it is the Role Identifier of the Role Assignment being deleted. If this value is NOT NULL, but less than 0x40000000, it MUST be security change Type Flag 0x00000001, indicating a role inheritance change. If this value is NULL, it indicates a security principal has been removed from all roles on the security scope.</td>
</tr>
<tr>
<td>0x04000000</td>
<td>MUST be NULL.</td>
</tr>
</tbody>
</table>

Event Object Type = 0x00000002 (List)

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00001000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00002000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00004000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00008000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00020000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00080000</td>
<td>If the value is NULL, it indicates that the anonymous user permission is updated. If this value is greater than or equal to 0x40000000, it indicates that a role assignment is added. If the value is NOT NULL and less than 0x40000000, it is a Security Change Type Flag, and indicates that a role inheritance has changed.</td>
</tr>
<tr>
<td>0x02000000</td>
<td>If this value is NOT NULL and greater than or equal to 0x40000000, it is the Role Identifier of the Role Assignment being deleted. If this value is NOT NULL, but less than 0x40000000, it MUST be Security Change Type Flag 0x00000001, indicating a role inheritance change. If this value is NULL, it indicates a security principal has been removed from all roles on the security scope.</td>
</tr>
</tbody>
</table>

Event Object Type = 0x00000004 (Site)

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00001000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>Event Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>0x00002000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00004000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00008000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00040000</td>
<td>If this value is greater than or equal to 0x40000000, it specifies the Identifier of the Role which is added. Otherwise, this is a Security Change Type Flag.</td>
</tr>
<tr>
<td>0x00020000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00800000</td>
<td>Role Identifier.</td>
</tr>
<tr>
<td>0x01000000</td>
<td>Role Identifier.</td>
</tr>
<tr>
<td>0x02000000</td>
<td>If this value is NOT NULL and greater than or equal to 0x40000000, it is the Role Identifier of the Role Assignment being deleted. If this value is NOT NULL, but less than 0x40000000, it MUST be Security Change Type Flag 0x00000001, indicating a role inheritance change. If this value is NULL, it indicates a security principal has been removed from all roles on the security scope.</td>
</tr>
<tr>
<td>0x00080000</td>
<td>If the value is NULL, it indicates that the anonymous user permission is updated. If this value is greater than or equal to 0x40000000, it indicates that a role assignment is added. If the value is NOT NULL and less than 0x40000000, it is a Security Change Type Flag, and indicates that a role inheritance has changed.</td>
</tr>
<tr>
<td>0x08000000</td>
<td>MUST be NULL.</td>
</tr>
</tbody>
</table>

Event Object Type = 0x00000008 (Site Collection)

MUST be NULL.

Event Object Type = 0x00000010 (File)
MUST be NULL.

Event Object Type = 0x00000020 (Folder)
MUST be NULL.

Event Object Type = 0x00000040 (Alert Subscription)
MUST be NULL.

Event Object Type = 0x00000080 (Security Principal)

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00001000</td>
<td>MUST be NULL.</td>
</tr>
<tr>
<td>0x00002000</td>
<td>MUST be NULL, 0 or 1. If the value is 1, it indicates that the Security Principal's status as a site collection administrator has been changed. If the value is 0 or NULL, it has not been changed.</td>
</tr>
<tr>
<td>0x00004000</td>
<td>MUST be NULL.</td>
</tr>
</tbody>
</table>
Event Object Type = 0x00000100 (Security Group)
MUST be NULL.

Event Object Type = 0x00000200 (Content Type)
MUST be NULL.

Event Object Type = 0x00000400 (Field Template)
MUST be NULL.

Event Object Type = 0x00000800 (Security Policy)
MUST be NULL.

Event Object Type = 0x00001000 (View)
MUST be NULL.

2.2.1.11 Change Log SiteId

Change Log SiteId is a GUID and is part of a Change Log Entry. This data has different meanings based on the Event Object Type and Event Type data of the same Change Log Entry. This value is the site collection identifier of the Site Collection under which the event has occurred, except in the following cases, it is an empty GUID.

Event Object Type = 0x00000800 (Security Policy)

<table>
<thead>
<tr>
<th>Object Type</th>
<th>Event Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00002000</td>
<td>MUST be an empty GUID.</td>
</tr>
</tbody>
</table>

Event Object Type = 0x00000080 (Site Collection)

<table>
<thead>
<tr>
<th>Object Type</th>
<th>Event Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00020000</td>
<td>When this value is not Empty GUID, it is the site collection identifier is restored. If this value is empty GUID, the content database containing this change log was restored.</td>
</tr>
</tbody>
</table>

2.2.1.12 Change Log WebId

Change Log WebId is a GUID and is part of a Change Log Entry. This data has different meanings based on the Event Object Type and Event Type data of the same Change Log Entry. This value is the site identifier of the Site under which the Event has occurred, except in the following cases, it MUST be NULL.

<table>
<thead>
<tr>
<th>Object Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00000008</td>
</tr>
<tr>
<td>0x00000080</td>
</tr>
<tr>
<td>0x00000100</td>
</tr>
</tbody>
</table>
2.2.1.13 Page Navigation Dependency Examples

The following are Unicode string examples of page navigation dependencies:

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>P:C</td>
<td>1002</td>
</tr>
<tr>
<td>P:C</td>
<td>1006</td>
</tr>
<tr>
<td>P:C</td>
<td>1004</td>
</tr>
<tr>
<td>P:C</td>
<td>1003</td>
</tr>
<tr>
<td>P:C</td>
<td>1027</td>
</tr>
<tr>
<td>P:C</td>
<td>1025</td>
</tr>
<tr>
<td>P:C</td>
<td>1026</td>
</tr>
</tbody>
</table>

2.2.2 Bit Fields and Flag Structures

2.2.2.1 Event Object Type Flags

A 4-byte unsigned integer bit mask that specifies the type of object upon which an Event had happened. The only valid values of the Event Object Type Flags bits are specified, as follows:

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00000001</td>
<td>The Event is associated with a List Item.</td>
</tr>
<tr>
<td>0x00000002</td>
<td>The Event is associated with a List.</td>
</tr>
<tr>
<td>0x00000004</td>
<td>The Event is associated with a Site.</td>
</tr>
<tr>
<td>0x00000008</td>
<td>The Event is associated with a Site Collection.</td>
</tr>
<tr>
<td>0x00000010</td>
<td>The Event is associated with a File.</td>
</tr>
<tr>
<td>0x00000020</td>
<td>The Event is associated with a Folder.</td>
</tr>
<tr>
<td>0x00000040</td>
<td>The Event is associated with an Alert.</td>
</tr>
<tr>
<td>0x00000080</td>
<td>The Event is associated with a user.</td>
</tr>
<tr>
<td>0x00000100</td>
<td>The Event is associated with a group (2).</td>
</tr>
<tr>
<td>0x00000200</td>
<td>The Event is associated with a Content Type.</td>
</tr>
<tr>
<td>0x00000400</td>
<td>The Event is associated with a Field.</td>
</tr>
<tr>
<td>Value</td>
<td>Meaning</td>
</tr>
<tr>
<td>-----------</td>
<td>----------------------------------------------</td>
</tr>
<tr>
<td>0x00000800</td>
<td>The Event is associated with a security policy.</td>
</tr>
<tr>
<td>0x00010000</td>
<td>The Event is associated with a View.</td>
</tr>
</tbody>
</table>

### 2.2.2.2 Event Type Flags

A 4 byte unsigned integer bit mask that specifies the type of an Event which can have one or more flags set. The only valid values of the Event Type flags bits are specified, as follows:

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00000001</td>
<td>A list item is added.</td>
</tr>
<tr>
<td>0x00000002</td>
<td>A list item is modified.</td>
</tr>
<tr>
<td>0x00000004</td>
<td>A list item is deleted.</td>
</tr>
<tr>
<td>0x00000008</td>
<td>A list item is restored from a backup.</td>
</tr>
<tr>
<td>0x00000010</td>
<td>A discussion list item is added.</td>
</tr>
<tr>
<td>0x00000020</td>
<td>A discussion list item is modified.</td>
</tr>
<tr>
<td>0x00000040</td>
<td>A discussion list item is deleted.</td>
</tr>
<tr>
<td>0x00000080</td>
<td>A discussion list item is closed.</td>
</tr>
<tr>
<td>0x00000100</td>
<td>A discussion list item is activated.</td>
</tr>
<tr>
<td>0x00001000</td>
<td>A generic add event.</td>
</tr>
<tr>
<td>0x00002000</td>
<td>A generic modification event.</td>
</tr>
<tr>
<td>0x00004000</td>
<td>A generic delete event.</td>
</tr>
<tr>
<td>0x00008000</td>
<td>A generic rename event.</td>
</tr>
<tr>
<td>0x00010000</td>
<td>Move into.</td>
</tr>
<tr>
<td>0x00020000</td>
<td>Restore.</td>
</tr>
<tr>
<td>0x00040000</td>
<td>A permission level is added.</td>
</tr>
<tr>
<td>0x00080000</td>
<td>A Role Assignment is added.</td>
</tr>
<tr>
<td>0x01000000</td>
<td>A modification executed by the system.</td>
</tr>
<tr>
<td>0x02000000</td>
<td>A member is added to a group (2).</td>
</tr>
<tr>
<td>0x04000000</td>
<td>A member is deleted from a group (2).</td>
</tr>
<tr>
<td>0x08000000</td>
<td>A permission level is deleted.</td>
</tr>
<tr>
<td>0x10000000</td>
<td>A permission level is updated.</td>
</tr>
<tr>
<td>0x20000000</td>
<td>A Role Assignment is deleted.</td>
</tr>
<tr>
<td>0x40000000</td>
<td>Move away.</td>
</tr>
</tbody>
</table>
### 2.2.2.3 Security Change Type Flags

A 32 bit mask that specifies modifications made to security settings. This Flag MUST be used in conjunction with an Event Type Flag. The valid values of the Security change Type Flag bits MUST be one of the following:

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00000000</td>
<td>No additional operation.</td>
</tr>
<tr>
<td>0x00000001</td>
<td>Remove Role Assignments on the current Security Scope and make it inherit Role Assignments from the parent Security Scope. This flag is meaningful only if the Event Type Flag is 0x02000000.</td>
</tr>
<tr>
<td>0x00000002</td>
<td>The current Site should define its own Roles, instead of inheriting them from the parent Site. This flag is meaningful only if the Event Type Flag is 0x00040000.</td>
</tr>
<tr>
<td>0x00000004</td>
<td>The current Security Scope should define its own Role Assignments, instead of inheriting them from the parent Security Scope. This flag is meaningful only if the Event Type Flag is 0x00080000.</td>
</tr>
<tr>
<td>0x00000008</td>
<td>Copy the Roles defined on the parent site to this Site. This flag is meaningful only if the Event Type Flag is 0x00040000.</td>
</tr>
<tr>
<td>0x00000010</td>
<td>Copy the Role Assignments defined on the parent Security Scope to this Security Scope. This flag is meaningful only if the Event Type Flag is 0x00040000 or 0x00080000.</td>
</tr>
</tbody>
</table>

### 2.2.2.4 Delete Flags

A 4-byte unsigned integer bit mask that specifies weather orphaned data is to be deleted for a Site. In rare circumstances, while a Site is being deleted, the deletion operation could be preempted or failed in midstream. As a result, data for the Site could remain in the database. This remaining data is often referred to as orphaned data because it does not have context without the existence of its Site. The only valid values of the Delete Flags bits are specified, as follows:

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Do not delete orphaned data if the Site's deletion operation was preempted or failed midstream.</td>
</tr>
<tr>
<td>8</td>
<td>Delete orphaned data if the Site's deletion operation was preempted or failed midstream.</td>
</tr>
</tbody>
</table>

### 2.2.2.5 Document Flags

A 4-byte unsigned integer bit mask providing Metadata about the Document. This can have one or more flags set. The only valid values of the document flags bits are specified as follows:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00000001</td>
<td>This Document contains dynamic content that SHOULD be sent through the CAML</td>
</tr>
<tr>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>interpreter, an implementation-specific dynamic content generation component. An example of this would be a Category <strong>Web bot</strong> present in the source of the Page.</td>
</tr>
<tr>
<td>0x00000002</td>
<td>The Document is a &quot;sub image&quot; of another Document. This is strongly correlated to the ExcludedType value in the security enumeration, and is set if this is an automatically generated thumbnail or Web image based on another item in the store.</td>
</tr>
<tr>
<td>0x00000004</td>
<td>The Document is a type for which there was a registered parser available at the time it was saved. A parser is an implementation-specific component that can extract data and Metadata from a Document, which can then be used to build a list of hyperlinks and <strong>Fields</strong> for <strong>Content Types</strong>.</td>
</tr>
<tr>
<td>0x00000008</td>
<td>The Document is a type which can contain hyperlinks.</td>
</tr>
<tr>
<td>0x00000010</td>
<td>The Document has an associated resource in the &quot;_private&quot; <strong>Folder</strong> that should be renamed in parallel when this file is renamed. An example of this is the count file for a hit counter <strong>Web Bot</strong>.</td>
</tr>
<tr>
<td>0x00000020</td>
<td>The Document is currently <strong>checked out</strong> to a user.</td>
</tr>
<tr>
<td>0x00000040</td>
<td>The Document is <strong>customized (1)</strong>.</td>
</tr>
<tr>
<td>0x00000080</td>
<td>The Page contains <strong>Web Parts</strong>. Defaults to a <strong>personal view</strong> (showing Web Parts that are specific to the user that browsed to the Page).</td>
</tr>
<tr>
<td>0x00000100</td>
<td>The Document is a type which can have a binary stream.</td>
</tr>
<tr>
<td>0x00000200</td>
<td>The Document is currently <strong>Checked Out</strong> to a location on the user's client system.</td>
</tr>
<tr>
<td>0x00000400</td>
<td>The Document has child Documents created by the Document transformations feature.</td>
</tr>
<tr>
<td>0x00000800</td>
<td>The Document is only a namespace entry for a List Item. (in other words it corresponds to a list item in a list (1) that should be filtered out from file system-centric enumerations).</td>
</tr>
<tr>
<td>0x00001000</td>
<td>Unused.</td>
</tr>
<tr>
<td>0x00002000</td>
<td>The Document has properties in its Metadata defining a custom order of the Content Types. This is valid only for Folders.</td>
</tr>
<tr>
<td>0x00004000</td>
<td>The Document <strong>SHOULD</strong> be customized (1) when &quot;undirtied&quot; (in other words, when dependency updates are performed for the Document). This is used for Documents such as a Document Library template, which is <strong>provisioned</strong> as <strong>uncustomized</strong> but <strong>SHOULD</strong> be customized (1) to denote Content Type information about the containing Document Library whenever that information is updated.</td>
</tr>
<tr>
<td>0x00008000</td>
<td>Used when a zero-byte Document is saved to a Document Library with required check out and at least one required Field. This is common in migration scenarios or with the use of older versions of the Windows® WebDAV redirector against the Windows SharePoint Service WebDAV implementation.</td>
</tr>
<tr>
<td>0xFFFF0000</td>
<td>Currently unused and <strong>SHOULD</strong> be ignored.</td>
</tr>
</tbody>
</table>

### 2.2.3 Binary Structures

No common binary structures are defined in this protocol.
2.2.4 Result Sets

No common result sets are defined in this protocol.

2.2.5 Tables and Views

2.2.5.1 AllUserData Table

Specified in [MS-WSSFO] section 2.2.7.3.

2.2.5.2 NameValuePair Table

The NameValuePair Table stores list item data for indexed fields. The NameValuePair Table is defined using Transact-Structured Query Language (T-SQL) syntax, as follows:

```sql
TABLE GroupMembership(
    SiteId    uniqueidentifier NOT NULL,
    WebId     uniqueidentifier NOT NULL,
    ListId    uniqueidentifier NOT NULL,
    ItemId    int NOT NULL,
    Level     tinyint DEFAULT 1 NOT NULL,
    FieldId   uniqueidentifier NOT NULL,
    Value     sql_variant
);
```

**SiteId:** The site collection identifier of the site collection containing the list item.

**WebId:** The site identifier of the site containing the list item.

**ListId:** The list identifier of the list containing the list item.

**ItemId:** The item identifier of the list item.

**Level:** The publishing level of the list item.

**FieldId:** The field identifier of an indexed field of the list item.

**Value:** The value of the indexed field specified by the FieldId column.

2.2.5.3 NameValuePair_Latin1_General_CI_AS Table

The NameValuePair_Latin1_General_CI_AS Table stores textual list item data for indexed fields using the Latin1_General_CI_AS collation order. The NameValuePair_Latin1_General_CI_AS Table is defined using T-SQL syntax, as follows:

```sql
TABLE GroupMembership(
    SiteId    uniqueidentifier NOT NULL,
    WebId     uniqueidentifier NOT NULL,
    ListId    uniqueidentifier NOT NULL,
    ItemId    int NOT NULL,
    Level     tinyint DEFAULT 1 NOT NULL,
    FieldId   uniqueidentifier NOT NULL,
    Value     nvarchar(255) COLLATE Latin1_General_CI_AS
);
```

---

[MS-WSSDLIM] — v20120630
Windows SharePoint Services: Content Database Document and List Item Management Communications Protocol Specification

Copyright © 2012 Microsoft Corporation.

Release: July 16, 2012
SiteId: The site collection identifier of the site collection containing the list item.

WebId: The site identifier of the site containing the list item.

ListId: The list identifier of the list containing the list item.

ItemId: The item identifier of the list item.

Level: The publishing level of the list item.

FieldId: The field identifier of an indexed field of the list item.

Value: The value of the indexed field specified by the FieldId column.

2.2.5.4 Collated NameValuePair Tables

A table exists for each collation order specified in [MS-WSSFO], section 2.2.3.4. These tables are identical to the NameValuePair_Latin1_General_CI_AS Table except that every reference to Latin1_General_CI_AS is replaced with the appropriate collation order name. For example, the NameValuePair_Albanian_CI_AS table uses the Albanian_CI_AS collation order for the Value column.

2.2.6 XML Structures

No common XML structures are defined in this protocol.

2.2.6.1 Namespaces

This specification defines and references various XML namespaces using the mechanisms specified in [XMLNS]. Although this specification associates a specific XML namespace prefix for each XML namespace that is used, the choice of any particular XML namespace prefix is implementation-specific and not significant for interoperability.

2.2.6.2 Simple Types

This specification does not define any common XML Schema simple type definitions.

2.2.6.3 Complex Types

This specification does not define any common XML Schema complex type definitions.

2.2.6.4 Elements

This specification does not define any common XML Schema element definitions.

2.2.6.5 Attributes

This specification does not define any common XML Schema attribute definitions.

2.2.6.6 Groups

This specification does not define any common XML Schema group definitions.

2.2.6.7 Attribute Groups

This specification does not define any common XML Schema attribute group definitions.
3 Protocol Details

3.1 WSSDLIM Server Details

3.1.1 Abstract Data Model

This section describes conceptual models 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.

3.1.1.1 Category Operations

Categories are changed by the front-end Web server using the following four stored procedures:

- proc_AddCategoryToWeb
- proc_DeleteCategory
- proc_AddDocToCategory
- proc_ListDocsInCategory

In addition, the Categories to which a document belongs are in the set of metadata for that document. This can be accessed using the proc_ReturnWebMetainfo stored procedure. Also, old Categories for a Document are deleted using the proc_UpdateDocument stored procedure.

3.1.1.1.1 Default Categories

The front-end Web server MUST create an initial set of Categories for every Site that is created. The set MUST contain the following Categories:

- Business
- Competition
- Expense Report
- Goals/Objectives
- Ideas
- In Process
- Miscellaneous
- Planning
- Schedule
- Travel
- VIP
- Waiting
These Categories MUST be localized into the language of the Site.

3.1.1.2 Change Log Operations

The protocol server stores a hierarchy of objects. The protocol server also maintains a Change Log table that records various Events that happened on those objects. The Events can be added directly by a protocol client using this protocol, or they can also be added indirectly when the client communicates with the server using a different protocol. For example, the client calls proc_AddDocument from [MS-WSSFO] which will result in an Event being added to the change. In addition, the client can use this protocol to retrieve information about Events that are currently in the Change Log. This documents specifies different types of events and how the information returned by the Back-End Database Server should be interpreted.

3.1.1.3 Publish and Un-publish Operations

The back-end database server stores a collection of documents. Each document can exist in up to three different publishing levels: Checked Out, Draft, and published, each with their own copy of the document and associated information. A document in the Draft publishing level typically has restricted visibility compared to a document in the Published publishing level. As part of this protocol, a front-end Web server can publish a document to change the document's current version from draft to published, or undo a publish to take the current version from Published to Draft. The following diagram illustrates this process.

![Figure 2: Publish and unpublish operations](image)

3.1.1.4 Check In and Check Out Operations

The front-end Web server can update the back-end database server to set a document's current version to Checked Out, which creates a separate logical copy of that document and associated information in the Checked Out publishing level. As part of this check out operation, the back-end database server stores a user identifier for that document recording the user that has that document checked out. Only that user can view the copy of the document that is in the Checked Out publishing level. If the check out operation is a short-term check out, the back-end database server also stores a time at which the checkout expires. The following diagram illustrates this process.
3.1.1.5 Historical Versioning Operations

The back-end database server maintains a (possibly empty) collection of historical versions for each document, containing information associated with previous revisions to the document. As part of this protocol, the front-end Web server can enumerate the collection of historical versions stored on the back-end database server as illustrated in the following diagram.

Figure 4: Historical versioning operations

The front-end Web server can also use this protocol to delete historical versions on the back-end database server or mark them as deleted in the Recycle Bin. As part of checking in a document, the front-end Web server can create a new historical version, copying the current version of the document and adding it to the collection of historical versions maintained by the back-end database server.

3.1.1.6 Link Fixup Operations

When a protocol client starts a link fixup operation, it starts the operation with a call to proc_StartUndirtyList call and retrieves the data columns from the AllUserData table that correspond to fields whose type allows forward links to be discovered for rows whose corresponding entry in the AllDocs table. Given the resulting range, the protocol client calls proc_GetListDataLinks to obtain the correct forward links for the data. The protocol client then computes the correct values for the field data and commits the data back to the AllUserData table, also calling proc_UndirtyListItem for each item. After repeating this procedure for every dirty list item, the protocol client includes a call to proc_FinishUndirtyList to complete the operation.
3.1.1.7 Theme Operations

Theme data for this protocol is maintained on both the front-end Web server and Back-End Database Server. The front-end Web server stores the content of the Theme Files in its file system. The Back-End Database Server stores additional Theme Metadata in one or more Content Databases. The Theme Metadata stored in the appropriate Content Databases is created and maintained when Theme data is loaded and applied to a Site.

Client: The Client referred to in the previous diagram is the Computer that will be requesting a Page for a Site from the front-end Web server.

Web Browser: A software application capable of displaying HTML Pages requested by the front-end Web server.

Theme: A collection of graphics and Cascading Style Sheets (CSS) that determine visual aspects of Pages for a Site. For example, a Theme can determine the background color of a Page, the Page's text color, font, and alignment, how hyperlinks on Pages change in behavior when clicked on by a mouse, and the presence of tooltips when a mouse is hovered over text or images on Pages.
Theme Files: Consists of images and CSS files that comprise a Theme. These files are stored on the front-end Web server File System and their contents are retrieved when the Client's Web Browser when requesting a Page from the front-end Web server.

Theme Name: A Unicode string that uniquely identifies the set of Theme Files for a particular Theme.

Theme Metadata: The Theme Metadata is comprised of information about the Theme Files stored on the front-end Web server. The Theme Metadata is stored in the Content Database on the Back-End Database Server. Theme Metadata consists of:

- The URL in of the Theme Files in Store-Relative Form.
- The Theme name.
- The installation path where Theme Files can be found on the File System of the front-end Web server.

Theme Installation Path: The directory path fragment where the Theme Files can be found on the front-end Web server File System for a particular Theme. For example, the Theme Files for the 'Wheat' Theme would found at 'Program Files\Common Files\Microsoft Shared\Web Server Extensions\12\template\themes\wheat'.

Theme XML File: The Theme XML File is a XML file located on the File System of the front-end Web server. For example, if the Theme XML File is translated into the 1033 language code identifier (LCID), then the file would be located at 'Program Files\Common Files\Microsoft Shared\Web Server Extensions\12\layouts\1033\spthemes.xml'.

Theme XSD File: The Theme XML File conforms to the XML schema definition (XSD) defined by the Theme XSD File. This file is located on the front-end Web server File System at the same location as that of the Theme XML File where LCID is the Language Code Identifier (LCID) that the Theme XSD File has been translated into. For example, if the Theme XSD File is translated into the 1033 Language Code Identifier (LCID), then the file would be located at 'Program Files\Common Files\Microsoft Shared\Web Server Extensions\12\layouts\1033\spthemes.xsd'.

3.1.1.8 Wide List Operations

Wide Lists are created when the number of fields required by a List of a specific type exceeds a fixed maximum per type allowed per Row in the Content Database.

The following table lists the number of fields allowed, per type of field, per row:

<table>
<thead>
<tr>
<th>Field Type</th>
<th>Allowable Fields</th>
</tr>
</thead>
<tbody>
<tr>
<td>nvarchar(255)</td>
<td>64</td>
</tr>
<tr>
<td>Ntext</td>
<td>32</td>
</tr>
<tr>
<td>sql_variant</td>
<td>8</td>
</tr>
<tr>
<td>Int</td>
<td>16</td>
</tr>
<tr>
<td>Float</td>
<td>12</td>
</tr>
<tr>
<td>datetime</td>
<td>8</td>
</tr>
<tr>
<td>Bit</td>
<td>16</td>
</tr>
<tr>
<td>Uniqueidentifier</td>
<td>1</td>
</tr>
</tbody>
</table>
3.1.2 Timers

An execution timeout timer on the protocol server governs the execution time for the client's requests. The amount of time is specified by a timeout value that is configured on the protocol server for all connections.

3.1.3 Initialization

A connection that uses the underlying protocol layers that are specified in Section 1.4 MUST be established before using this protocol as specified in [MS-TDS].

3.1.4 Message Processing Events and Sequencing Rules

The T-SQL syntax for each Stored Procedure and Result Set, and the variables they are composed of, is defined in the [MSDN-TSQL-Ref] protocol. In the T-SQL syntax, the variable name is followed by the type of the variable which can optionally have a length value in brackets and can optionally have a default value indicated by an equals sign followed by the default value. Unless otherwise specified, all Stored Procedures defined in this section are located in the Content Database.

For definitional clarity, a name has been assigned to any columns in the Result Sets that do not have a defined name in their current implementation. This does not affect the operation of the Result Set, as the ordinal position of any column with no defined name is expected by the front-end Web server. Such names are designated in the text using curly braces in the form \{name\}.

3.1.4.1 proc_AddCategoryToWeb

The proc_AddCategoryToWeb Stored Procedure is called to associate a Category to a Site. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_AddCategoryToWeb(
    @WebId       uniqueidentifier,
    @Category    nvarchar(128)
);
```

@WebId: The Site Identifier of the Site which will be associated with the specified category.

@Category: Category associated with the specified Site.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST NOT return any Result Sets.

3.1.4.2 proc_AddDependency

The proc_AddDependency Stored Procedure is called to create a document dependency between Documents. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_AddDependency(
    @SiteId     uniqueidentifier,
    @FullUrl    nvarchar(260),
    @Level      tinyint,
    @DepType    tinyint,
    @DepDesc    nvarchar(270)
);
```

[MS-WSSDLIM] — v20120630
Windows SharePoint Services: Content Database Document and List Item Management Communications Protocol Specification

Copyright © 2012 Microsoft Corporation.

Release: July 16, 2012
@SiteId: The Site Collection Identifier of the Site Collection which contains the specified Document.

@FullUrl: The URL of the Document in Store-Relative Form to which the Document Dependency will be created.

@Level: The Publishing Level of the specified Document.

@DepType: The dependency type. The following values are valid:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Document dependency. This updates items dependent on the specified document. The @DepDesc parameter is the store-relative form URL of the document that has changed.</td>
</tr>
<tr>
<td>3</td>
<td>Configuration dependency. This updates items dependent on changes to system configuration metadata, as specified in [MC-FPSEWM] section 2.2.2.3. The @DepDesc parameter is the metadata key for the metadata that has changed.</td>
</tr>
<tr>
<td>4</td>
<td>Navigation dependency. This updates items dependent on changes to navigation structures. The @DepDesc parameter contains the Web-Navigation-URL, as specified in [MC-FPSEWM] section 2.2.2.34, for a navigation structure node.</td>
</tr>
<tr>
<td>7</td>
<td>Usage dependency. This updates items dependent on changes to site usage statistics. The @DepDesc parameter is the store-relative form URL of the site.</td>
</tr>
</tbody>
</table>

@DepDesc: The dependency description, which varies according to the value of @DepType, as described in the preceding table.

Return Code Values: An integer which the protocol client MUST ignore.

Result Sets: MUST NOT return any Result Sets.

3.1.4.3 proc_AddDocToCategory

The proc_AddDocToCategory Stored Procedure is called to assign a Document to a category. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_AddDocToCategory(
    @DocId uniqueidentifier,
    @WebId uniqueidentifier,
    @Category nvarchar(128)
);
```

@DocId: The Document Identifier of the Document to be assigned to the category provided by the @Category Stored Procedure parameter.

@WebId: The Site Identifier of the Site which contains the specified Document.

@Category: The category name that the specified Document will be assigned.

Return Code Values: An integer which the protocol client MUST ignore.

Result Sets: MUST NOT return any Result Sets.
3.1.4.4 proc_AddEventToCache

The proc_AddEventToCache Stored Procedure is called to add an Event to the Back-End Database Server. The T-SQL syntax for the Stored Procedure is as follows.

```sql
PROCEDURE proc_AddEventToCache(
    @SiteId            uniqueidentifier,
    @WebId             uniqueidentifier,
    @ListId            uniqueidentifier,
    @ItemId            int,
    @ItemName          nvarchar(255),
    @ItemFullUrl       nvarchar(260),
    @DocId             uniqueidentifier,
    @EventType         int,
    @ModifiedBy        nvarchar(255),
    @TimeLastModified  datetime,
    @EventData         image,
    @ACL               image,
    @ScopeId           uniqueidentifier = NULL
);
```

@SiteId: The Site Collection Identifier of the Site Collection which contains the specified Event.

@WebId: The Site Identifier of the Site in which specified Event has occurred.

@ListId: The List Identifier of the List to which the Event is related.

@ItemId: The list item identifier from the List specified by @ListId that is associated with the Event. Its value MUST be -1 or a List Item Identifier. When its value is -1, the Stored Procedure MUST use @ItemFullUrl to determine which List Item is associated with this Event.

@ItemName: A string that represents the name of the List Item chosen by the application or NULL if the application does not use this information.

@ItemFullUrl: This parameter SHOULD use Store-Relative Form to the List Item associated with the Event, or the Store-Relative Form to the Site. This parameter MUST be NULL if the application specifies the List Item using @ItemId.<5>

@DocId: The Document Identifier of the Document associated with the List Item specified by either @ItemId or @ItemFullUrl. If this parameter is NULL, the stored procedure will compute @docId from the @ItemId and @ListId parameters.

@EventType: An integer that specifies Event Type Flags.

@ModifiedBy: A string which specifies the login name of a Security Principal who added this Event.

@TimeLastModified: A timestamp in Coordinated Universal Time (UTC) that specifies the time when this Event occurred.

@EventData: This parameter contains implementation-specific Event data.

@Acl: A byte array in the access control list (ACL) format. If this parameter is NULL, this stored procedure will use the Acl defined on the Security Scope specified by the @ScopeId parameter. If the ScopeId parameter is also NULL, the stored procedure will use the ACL from the object specified by @ItemFullUrl.
@ScopeId: A scope identifier which specifies the Security Scope.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST NOT return any Result Sets.

3.1.4.5 proc_AddGhostDocument

The proc_AddGhostDocument stored procedure is called to create an uncustomized file. The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_AddGhostDocument(
    @SiteId                   uniqueidentifier,
    @WebId                    uniqueidentifier,
    @DocId                    uniqueidentifier,
    @DocDirName               nvarchar(256),
    @DocLeafName              nvarchar(128),
    @Level                    tinyint,
    @UIVersion                int,
    @EnableMinorVersions      bit,
    @DocSize                  int,
    @DocFlags                 int,
    @OnRestore                bit = 0,
    @Overwrite                bit OUTPUT,
    @UserId                   int,
    @HasDeleteListItemsRight  bit,
    @SetupPathVersion         tinyint,
    @SetupPath                nvarchar(255),
    @SetupPathUser            nvarchar(255),
    @ListId                   uniqueidentifier = NULL,
    @DoclibRowId              int = NULL,
    @fCheckQuotaAndWriteLock  bit = 0,
    @DTM                      datetime = NULL OUTPUT
);
```

@SiteId: The site collection identifier of the site collection that contains the site.

@WebId: The site identifier of the site that contains the uncustomized file.

@DocId: The document identifier for the new uncustomized file to be created. An existing file with the specified document identifier MUST NOT exist.

@DocDirName: The directory name for the new uncustomized file.

@DocLeafName: The leaf name for the new uncustomized file.

@Level: The publishing level for the new uncustomized file. Refer to [MS-WSSFO], section 2.2.2.6.

@UIVersion: The user interface (UI) version number for the new uncustomized file.

@EnableMinorVersions: This parameter is not used and MUST be ignored.

@DocSize: The size, in bytes, of the uncustomized file.

@DocFlags: The document flag, as specified in section 2.2.5, for the uncustomized file to be created.
@OnRestore: Whenever an uncustomized file is successfully created, an event object type flag, as specified in section 2.2.2.1, of "0x00000010" with an event type flag, as specified in section 2.2.2.2, of "0x00001000" is recorded in the change log. Along with this, the Datetime in Coordinated Universal Time (UTC) that the uncustomized file was created is also recorded in the change log. If this parameter is "1", an additional event object type flag of "0x00000010" with the event type flag of "0x00100000" is recorded in the change log and the Datetime recorded is NULL. No Datetime is recorded in the change log. If this parameter is set to something other than "1", the Datetime recorded is the current UTC.

@Overwrite: If this parameter is set to a value other than "1" on input, this parameter MUST be ignored. However, if it is set to "1" on input and an existing file is found specified by the @DocDirName and @DocLeafName parameters and the @HasDeleteListItemsRight parameter is set to "1", the existing file is deleted and replaced with the new uncustomized file being created. If the existing file is successfully deleted, the @Overwrite parameter is set to "1" on output. If the existing file is not successfully deleted, the @Overwrite parameter is set to zero ("0") on output.

@UserId: The user identifier of the user requesting the operation. If the specified @Level parameter is set to the publishing level of "draft", the draft owner for the file is set to this user identifier. If the specified @Level parameter is set to the publishing level of "checked out", the checked out owner for the file is set to this user identifier.

@HasDeleteListItemsRight: This parameter is only used if the @Overwrite parameter is set to "1" on input. When set to "1", this parameter’s schematics are described in the definition for the @Overwrite parameter.

@SetupPathVersion: This parameter specifies the directory path fragment where the uncustomized file content is located on the front-end Web server's file system. This parameter MUST be one of the following values:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;2&quot;</td>
<td>The @SetupPath parameter value supplied is relative to the install location of wss2 on the front-end Web server (for example Program Files\Common Files\Microsoft Shared\Web Server Extensions\60).</td>
</tr>
<tr>
<td>&quot;3&quot;</td>
<td>The @SetupPath parameter value supplied is relative to the install location of wss3 on the front-end Web server (for example Program Files\Common Files\Microsoft Shared\Web Server Extensions\12).</td>
</tr>
</tbody>
</table>

@SetupPath: This specifies the Unicode string directory path fragment relative to the base directory path specified by the @SetupPathVersion parameter. Taken together, the @SetupPath and @SetupPathVersion parameters specify where the uncustomized file can be found on the front-end Web server's file system.

@SetupPathUser: This specifies the Unicode string of the login name of the user that is creating the uncustomized file.

@ListId: This parameter is optional and defaults to NULL if not specified. If this parameter is not NULL, it MUST be the list identifier of a list (1) contained within the specified site. If this parameter is NULL, the uncustomized file to be created is not contained within a list (1).

@DoclibRowId: This parameter is optional and defaults to NULL if not specified. If this parameter is not NULL, it MUST be the Document Library Row Identifier for the new uncustomized file. If the uncustomized file is not contained within a list, this parameter and the @ListId parameter MUST be NULL.
@fCheckQuotaAndWriteLock: This parameter is optional and defaults to zero ("0") if not specified. If this parameter is set to zero ("0"), it MUST be ignored. However, if it is set to a value other than zero ("0"), the stored procedure verifies that adding the uncustomized file will not exceed the specified site collection's site collection quota and that the site collection's site collection flag does not contain the WRITELOCK ("0x00000001") bit. Refer to [MS-WSSFO], section 2.2.2.9.

@DTM: This parameter is optional and defaults to NULL if not specified. If specified, the input value is ignored and the value set on output is the Datetime in UTC when the uncustomized file was created and when the uncustomized file was last modified.

Return Code Values: An integer which MUST be listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;0&quot;</td>
<td>Successful execution.</td>
</tr>
</tbody>
</table>
| "3"   | This return code MUST be returned if any of the following are true:  
|       | - If the site collection specified by the @SiteId parameter does not exist.  
|       | - If the site specified by the @WebId parameter does not exist.  
|       | - The site collection specified by the @SiteId parameter exists, but its directory name does not match the value specified by the @DocDirName parameter. |
| "5"   | This return code is returned if the uncustomized file failed to be created because the user identifier specified by the @UserId parameter does not have the appropriate permissions. This return code is also returned if the @Overwrite parameter is set to "1" and the @HasDeleteListItemsRight parameter is set to "1" and an existing file was not found specified by the @DocDirName and @DocLeafName parameters. |
| "80"  | The uncustomized file failed to be created because an existing file was found specified by the @DocDirName and @DocLeafName parameters and the @Overwrite parameter was set to a value other than "1". |
| "212" | If the @fCheckQuotaAndWriteLock parameter is set to any value other than zero ("0"), the creation of the uncustomized file fails with this return code if the specified site collection has its WRITELOCK ("0x00000001") site collection flag bit set. Refer to [MS-WSSFO], section 2.2.2.9. |
| "1816"| If the @fCheckQuotaAndWriteLock parameter is set to any value other than zero ("0"), the creation of the uncustomized file fails with this return code if adding the uncustomized file exceeds the quota for the specified site collection. |

Result Sets: MUST NOT return any result sets.

3.1.4.6 proc_AddNewRowOrdToList

The proc_AddNewRowOrdToList Stored Procedure is called to allocate one new row in AllUserData per List Item for a Wide List. The T-SQL syntax for the Stored Procedure is as follows:

```
PROCEDURE proc_AddNewRowOrdToList(
    @SiteId    uniqueidentifier,
    @WebId     uniqueidentifier,
    @ListID    uniqueidentifier,
    @RowOrdinal int,
    @CheckSchemaVersion int NULL
);"
@SiteId: The Site Collection Identifier of the Site Collection which contains the specified Site.

@WebId: The Site Identifier of the Site which contains the specified List.

@ListID: The List Identifier of the List for which the new row ordinal is added.

@RowOrdinal: A number indicating the next row ordinal to use for this List. This number MUST be at most one greater than the current maximum row ordinal for the list.

@CheckSchemaVersion: The List schema version number to check against to ensure no intervening change to the List schema was made.

Return Code Values: An integer which MUST be listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful execution.</td>
</tr>
<tr>
<td>87</td>
<td>Updating the List could not be finished because of resource constraints.</td>
</tr>
<tr>
<td>1638</td>
<td>The List schema version has changed, and the operation cannot continue.</td>
</tr>
</tbody>
</table>

Result Sets: MUST NOT return any Result Sets.

3.1.4.7 proc_AddNewRowOrdToListItem

The proc_AddNewRowOrdToListItem Stored Procedure is called to add one row for a List Item in a Wide List. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_AddNewRowOrdToListItem(
    @SiteId    uniqueidentifier,
    @WebId     uniqueidentifier,
    @ListID    uniqueidentifier,
    @ItemID    int,
    @RowOrdinal int,
    @CheckSchemaVersion int = NULL
);
```

@SiteId: The Site Collection Identifier of the Site Collection which contains the specified Site.

@WebId: The Site Identifier of the Site which contains the specified List.

@ListID: The List Identifier of the List for which the new row ordinal is added.

@ItemID: The integer identifier of the List Item for which to add new row ordinal support.

@RowOrdinal: A number indicating the ordinal of the row to add for this List Item. This number MUST be greater than 1 and less than the current maximum row ordinal for the List.

@CheckSchemaVersion: The List schema version number to check against to ensure no intervening change to the List schema was made.

Return Code Values: An integer which MUST be listed in the following table:
### Value | Description
--- | ---
0 | Successful execution.
87 | Updating the List could not be finished because of resource constraints.
1638 | The List schema version has changed, and the operation cannot continue.

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

#### 3.1.4.8 proc_AL

The proc_AL Stored Procedure is called to store a Link from a Document, List, Folder, or Document Library to another Document, List, Folder, or Document Library. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_AL(
    @SiteId uniqueidentifier,
    @DirName nvarchar(256),
    @LeafName nvarchar(128),
    @Level tinyint,
    @LinkNumber int,
    @TargetDirName nvarchar(256),
    @TargetLeafName nvarchar(128),
    @Type tinyint,
    @Security tinyint,
    @Dynamic tinyint,
    @ServerRel bit,
    @Search ntext,
    @WP uniqueidentifier = NULL,
    @Fld uniqueidentifier = NULL
);
```

- **@SiteId:** The Site Collection Identifier of the Site Collection which contains the specified Document.
- **@DirName:** The directory name of the location of the Document that contains the Link to be stored. This MUST NOT be NULL.
- **@LeafName:** The leaf name of the Document that contains the Link. This MUST NOT be NULL.
- **@Level:** The Publishing Level of the source object.
- **@LinkNumber:** Ordinal number of the Link in the Document. This value MUST NOT be NULL.
- **@TargetDirName:** The directory name of the linked Document or List or Folder. This value MUST NOT be NULL.
- **@TargetLeafName:** The leaf name of the linked Document or List or Folder. This value MUST NOT be NULL.
- **@Type:** A one byte (tinyint) value represented as a single upper case ASCII character specifying the Link Type. The value MUST be one of the following:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>The Link is from the ACTION attribute of an HTML form tag.</td>
</tr>
<tr>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>B</td>
<td>The Link is from the attribute markup of a bot.</td>
</tr>
<tr>
<td>C</td>
<td>The Link is from an auto-generated table of contents. Agents MAY ignore the Link type when determining unreferenced files within a Site.</td>
</tr>
<tr>
<td>D</td>
<td>The Link references programmatic content, as in the HTML OBJECT or APPLET tags.</td>
</tr>
<tr>
<td>E</td>
<td>The Link is from a cascading style sheet (CSS).</td>
</tr>
<tr>
<td>F</td>
<td>The Link is from the SRC attribute of an HTML FRAME tag.</td>
</tr>
<tr>
<td>G</td>
<td>The Link is to a dynamic Web template for the containing Document.</td>
</tr>
<tr>
<td>H</td>
<td>The Link is from an HTML HREF attribute. This MAY also be used as a default Link Type value if a more precise type does not apply.</td>
</tr>
<tr>
<td>I</td>
<td>The Link is to a Document that the containing Document includes via an include Bot.</td>
</tr>
<tr>
<td>J</td>
<td>The Link is from a Field of this List Item.</td>
</tr>
<tr>
<td>K</td>
<td>Identical to 'H', except that the Link contains an HTML bookmark specifier.</td>
</tr>
<tr>
<td>L</td>
<td>The Link is a target in an HTML image map generated from an image map Bot.</td>
</tr>
<tr>
<td>M</td>
<td>The Link is to an image used in an HTML image map generated from an image map Bot.</td>
</tr>
<tr>
<td>O</td>
<td>The Link is part of a cross-page URL connection.</td>
</tr>
<tr>
<td>P</td>
<td>The Link is part of the markup of a URL within the source of the containing Document.</td>
</tr>
<tr>
<td>Q</td>
<td>The Link references a cascading style sheet (CSS) Document which provides style information for the containing Document.</td>
</tr>
<tr>
<td>R</td>
<td>The Link is from the master page File attribute of the @Page directive in the containing Document.</td>
</tr>
<tr>
<td>S</td>
<td>The Link is from an HTML SRC attribute.</td>
</tr>
<tr>
<td>T</td>
<td>The Link is to the index file used by a text search Bot on this Page.</td>
</tr>
<tr>
<td>V</td>
<td>The Link is based on the properties of the Document, rather than anything in the document stream. The Link type is used in tracking the Link between a Site and the master page URL used for the Site.</td>
</tr>
<tr>
<td>X</td>
<td>The Link is from an XML island within an HTML Document.</td>
</tr>
<tr>
<td>Y</td>
<td>The Link references an HTML Document whose HTML BODY tag attributes are used as a template for the attributes of the containing Document’s BODY tag.</td>
</tr>
<tr>
<td>Z</td>
<td>The Link is part of the markup of a URL which exists in a URL Zone in the containing Document, and is consequently not stored within the source of the containing Document.</td>
</tr>
</tbody>
</table>

@Security: Type of security for the Link. The value MUST be one of the following:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>H</td>
<td>The Link is to an Hypertext Transfer Protocol (HTTP) URL.</td>
</tr>
<tr>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>S</td>
<td>The Link is to an HTTPS URL.</td>
</tr>
<tr>
<td>T</td>
<td>The Link is to an SHTTP URL.</td>
</tr>
<tr>
<td>U</td>
<td>The Link transport security is unknown.</td>
</tr>
</tbody>
</table>

**@Dynamic:** A one byte (tinyint) value represented as a single upper case ASCII character which specifies the special Link Types. The value MUST be one of the following:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>The URL is static, which is the default, and requires no special handling.</td>
</tr>
<tr>
<td>D</td>
<td>The URL is dynamic, which is a Link to &lt;Site URL&gt;/_vti_bin/shtml.dll/DirName/LeafName. Such Links are used to call the FrontPage SmartHTML interpreter on a file.</td>
</tr>
<tr>
<td>L</td>
<td>The URL is to a layouts Page; that is, it contains a path segment with the string &quot;_layouts&quot;.</td>
</tr>
<tr>
<td>H</td>
<td>The URL is a history Link; that is, it contains a path segment with the string &quot;,_vti_history&quot;.</td>
</tr>
<tr>
<td>G</td>
<td>A nonabsolute Link from an uncustomized document that does not fall into any other category.</td>
</tr>
</tbody>
</table>

**@ServerRel:** @ServerRel is used to indicate if @TargetDirName and @TargetLeafName are used to describe the location of the Target Document, or whether only @TargetDirName stores the complete URL to the Target Document. This value MUST NOT be NULL.

**@Search:** Search terms to be used to surface this Link when performing a full-text search. This value MAY be an empty string, but it MUST NOT be NULL.

**@WP:** A Web Part identifier identifying the Web Part that is the source of the Link creation command.

**@Fld:** The Field Identifier of the Field that is the source of the Link definition.

**Return Code Values:** An integer which MUST be 0.

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

### 3.1.4.9 proc_CheckoutDocumentInternal

The proc_CheckoutDocumentInternal Stored Procedure is called to request or renew Short-Term Check Out, or request long-term check out on a Document. The T-SQL syntax for the Stored Procedure is as follows:

```t-sql
PROCEDURE proc_CheckoutDocumentInternal(
    @SiteId uniqueidentifier,
    @WebId uniqueidentifier,
    @DirName nvarchar(256),
    @LeafName nvarchar(128),
    @Level tinyint,
    @EnableMinorVersions bit,
    @IsModerated bit,
    @UserId int,
    @CheckoutTimeout int,
    @RefreshCheckout bit,
)
```
@SiteId: The Site Collection Identifier of the Site Collection which contains the specified Document to be checked out.

@WebId: The Site Identifier of the Site which contains the specified Document.

@DirName: The directory name of the Document.

@LeafName: The leaf name of the Document.

@Level: The Publishing Level of the Document. Refer to [MS-WSSFO], section 2.2.2.6, for valid values.

@EnableMinorVersions: A bit flag specifying whether the Document Library containing the Document has minor version numbering enabled. If Minor Version numbering is enabled for the Document Library containing the Document, this parameter MUST be set to "1"; otherwise this parameter MUST be set to "0". If the Document is not in a Document Library, this parameter MUST be set to "0". This parameter MUST NOT be NULL.

@IsModerated: A bit flag specifying whether the Document Library containing the Document has moderation enabled. If the Document Library containing the Document is a moderated object, this parameter MUST be set to "1"; otherwise this parameter MUST be set to "0". If the Document is not in a Document Library, this parameter MUST be set to "0". This parameter MUST NOT be NULL.

@UserId: The User Identifier for the current user who is requesting a Short-Term Check Out or a Long-Term Check Out on the Document. This value MUST refer to an existing User identifier for the specified Site Collection.

@CheckoutTimeout: The timeout in minutes for Short-Term Check Out on the Document. The @CheckoutTimeout parameter MUST be NULL if a Long-Term Check Out on the Document is being specified.

@RefreshCheckout: A bit flag specifying whether the Short-Term Check Out on the Document needs to be refreshed. If this parameter is set to "1", the existing Short-Term Check Out on the Document MUST be refreshed for the number of minutes specified by the @CheckoutTimeout parameter. This parameter MUST be set to "0" to request a new Short-Term Check Out or Long-Term Check Out on the Document. This parameter MUST NOT be NULL.

@CheckoutToLocal: A bit flag specifying whether the Document is to be copied to local storage on the user's computer for editing. If this parameter is set to "1", the User computer SHOULD make a local copy of the Document Stream for editing and proc_CheckoutDocumentInternal MUST NOT make a Checked Out version of the Document in the store.

@IsForceCheckout: A bit flag specifying whether the Document Library containing the Document requires Documents to be checked out before any changes can be made. If the Document Library containing the Document enforces Documents to be checked out before editing, this parameter MUST be set to "1". If the Document is not in a Document Library, this parameter MUST be set to "0". This parameter MUST NOT be NULL.

@Now: The current Coordinated Universal Time (UTC) time.

Return Code Values: An integer which MUST be listed in the following table:
<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful execution.</td>
</tr>
<tr>
<td>3</td>
<td>File not found. A Document corresponding to the specified @SiteId, @WebId, @DirName, @LeafName, and @Level parameters was not found.</td>
</tr>
<tr>
<td>33</td>
<td>Lock could not be acquired. Another User has already applied a Short-Term Check Out or a Long-Term Check Out on the Document.</td>
</tr>
<tr>
<td>87</td>
<td>Invalid Parameter. The List Item for the specified Document could not be added. One or more of the parameters @SiteId, @DirName, @LeafName, and @Level are incorrect.</td>
</tr>
<tr>
<td>154</td>
<td>Invalid Minor Version value. The Minor Version value for the Document would exceed the maximum allowed value (511) if the document was Checked Out.</td>
</tr>
<tr>
<td>158</td>
<td>Checkout required. The parameter @IsForceCheckout is set to &quot;1&quot; indicating that Documents have to be Checked Out before editing. But the specified Document is not.</td>
</tr>
<tr>
<td>160</td>
<td>One or more arguments are incorrect. Document is at Draft level and @UserId is NULL.</td>
</tr>
<tr>
<td>212</td>
<td>Site Collection locked. The operation could not be performed because the Site Collection containing the Document is in read-only mode.</td>
</tr>
<tr>
<td>1630</td>
<td>Unsupported Document type. The Document specified is not valid for Check Out; Folders and Sites cannot be Checked Out.</td>
</tr>
<tr>
<td>1816</td>
<td>Disk quota error. The Site Collection disk Quota has been reached.</td>
</tr>
</tbody>
</table>

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

### 3.1.4.10 proc_CloneDoc

The proc_CloneDoc Stored Procedure is called to create a copy or a new Version of the specified Document. The target Document has exactly the same contents and properties as that of the original Document except Publishing Level and Version. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_CloneDoc(  
    @SiteId                   uniqueidentifier,  
    @DirName                  nvarchar(256),  
    @LeafName                 nvarchar(128),  
    @NewInstanceID            int = NULL,  
    @NewItemID int = NULL OUTPUT,  
    @Now                      datetime = NULL,  
    @OldLevel                 int = NULL,  
    @NewLevel                 int = NULL,  
    @EnableMinorVersions      bit = NULL,  
    @IsModerated              bit = NULL,  
    @UserId int = NULL,  
    @NewLeafName              nvarchar(128) = NULL
);  
```

**@SiteId:** The Site Collection Identifier of the Site Collection which contains the specified Document.

**@DirName:** The directory name containing the existing Document.

**@LeafName:** The leaf name of the existing Document.
@NewInstanceId: If the Document is inside a Meeting Workspace site, this parameter MUST be the identifier of the meeting workspace site. Otherwise, the value can be omitted and it defaults to NULL.

@NewItemID: If the target Document is successfully created, proc_CloneDoc MUST return the identifier of the target document. If the operation fails, this value MUST be ignored.

@Now: The current Coordinated Universal Time (UTC) time. If the value is omitted it defaults to NULL.

@OldLevel: A Publishing Level | described in [MS-WSSFO], section 2.2.2.6 specifying the publishing status of the existing Document. If the value is omitted, it defaults to NULL. If the value equals @NewLevel or NULL, a new Document MUST be created.

@NewLevel: The Publishing Level described in [MS-WSSFO], section 2.2.2.6 of the target Document or Version. If the value is omitted, it defaults to NULL. If the value equals @OldLevel or NULL, a new Document MUST be created.

@EnableMinorVersions: A bit flag specifying whether the Document Library containing the Document has Minor Version numbering enabled. If Minor Version numbering is enabled for the Document Library containing the Document, this parameter MUST be set to "1"; otherwise this parameter MUST be set to "0". If the Document is not in a Document Library, this parameter MUST be set to "0". This parameter MUST NOT be NULL.

@IsModerated: A bit flag specifying whether the Document Library containing the Document has moderation enabled. If the Document Library containing the Document is a moderated object, this parameter MUST be set to "1"; otherwise this parameter MUST be set to "0". If the Document is not in a Document Library, this parameter MUST be set to "0". This parameter MUST NOT be NULL.

@UserId: The identifier for the current user who is requesting this operation. The value MUST be provided if @NewInstanceId is NULL, or if the values of @OldLevel is not the same as @NewLevel. If a value is provided, the value MUST refer to an existing user identifier for the specified Site Collection. If the value is omitted, it defaults to NULL.

@NewLeafName: The leaf name of the target Document or Version. If the value is omitted, it defaults to NULL.

**Return Code Values:** An integer which MUST be listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful execution. The document was successfully copied.</td>
</tr>
<tr>
<td>3</td>
<td>File Not Found. The target Document corresponding to the specified @SiteId, @WebId, @DirName, and @NewLeafName could not be created.</td>
</tr>
<tr>
<td>87</td>
<td>Invalid Parameter. One or more of the parameters, @SiteId, @WebId, @DirName, @LeafName, or @NewLevel, are not valid.</td>
</tr>
<tr>
<td>160</td>
<td>One or more arguments are incorrect. Value of @NewLevel is Draft and @UserID is NULL.</td>
</tr>
<tr>
<td>212</td>
<td>Site Collection locked. The operation could not be performed because the Site Collection containing the Document is in Read-Only Mode.</td>
</tr>
<tr>
<td>1816</td>
<td>Disk quota error. The quota for Site Collection has reached the maximum allowable limit.</td>
</tr>
</tbody>
</table>

**Result Sets:** MUST NOT return any Result Sets.
3.1.4.11  proc_ConvertJunctionToLookup

The proc_ConvertJunctionToLookup Stored Procedure is called to convert the type of a lookup field of a List from Multivalued lookup field to Single-valued lookup field. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_ConvertJunctionToLookup(
    @SiteId          uniqueidentifier,
    @ListId          uniqueidentifier,
    @FieldId         uniqueidentifier,
    @ColName         nvarchar(64),
    @RowOrdinal      int
);
```

- **@SiteId**: The Site Collection Identifier of the Site Collection which contains the specified List.
- **@ListId**: The List Identifier of the List containing the specified lookup field.
- **@FieldId**: The identifier of the specified lookup field.
- **@ColName**: The name of the column in the UserData view which corresponds to the specified lookup field.
- **@RowOrdinal**: It MUST be a 0-based ordinal of the row which contains the column corresponding to the specified lookup field. Additional rows are used when a List has more user-defined columns of one or more data types than can fit in a single row of this view.

**Return Code Values**: An integer which MUST be 0.

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

3.1.4.12  proc_ConvertLookupToJunction

The proc_ConvertLookupToJunction Stored Procedure is called to convert the type of a lookup field of a List from Single-valued lookup field to Multi-valued lookup field. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_ConvertLookupToJunction(
    @SiteId          uniqueidentifier,
    @ListId          uniqueidentifier,
    @FieldId         uniqueidentifier,
    @ColName         nvarchar(64),
    @RowOrdinal      int
);
```

- **@SiteId**: The Site Collection Identifier of the Site Collection which contains the specified List.
- **@ListId**: The List Identifier of the List containing the specified lookup field.
- **@FieldId**: The identifier of the specified lookup field.
- **@ColName**: The name of the column in the UserData view which corresponds to the specified lookup field.
@RowOrdinal: It MUST be a 0-based ordinal of the row which contains the column corresponding to the specified lookup field. Additional rows are used when a List has more user-defined columns of one or more data types than can fit in a single row of this view.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST NOT return any Result Sets.

3.1.4.13 proc_CopyUrl

The proc_CopyUrl Stored Procedure is called to copy a Site Collection, or subsite to a new location specified by a new URL. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_CopyUrl(
    @SiteId         uniqueidentifier,
    @SubWebId       uniqueidentifier,
    @OldUrl         nvarchar(260),
    @NewUrl         nvarchar(260),
    @UserId         int,
    @RenameFlags    int = 0,
    @PutFlags       int = 0,
    @ReturnFlags    int = 0,
    @AttachmentOp   int = 3,
    @ParseDocsNow   tinyint = NULL OUTPUT,
    @FailedUrl      nvarchar(260) = NULL OUTPUT
);
```

@SiteId: The Site Collection Identifier of the Site Collection.

@SubWebId: A Site Identifier that uniquely identifies the SubSite. MUST NOT be NULL.

@OldUrl: Current URL of the item being copied.

@NewUrl: URL which is to replace the @OldUrl.

@UserId: User Identifier of the requester.

@RenameFlags: A 4-byte integer bit mask determining the object rename options. This MAY have one or more flags set. The default value is 0, but it MUST NOT be NULL. The valid flags are:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00000000</td>
<td>Default behavior: Rename all dependent items.</td>
</tr>
<tr>
<td>0x00000001</td>
<td>Do not update all related Documents. This flag is no longer used in the current version.</td>
</tr>
<tr>
<td>0x00000002</td>
<td>Create directories if they do not exist.</td>
</tr>
<tr>
<td>0x00000004</td>
<td>Server SHOULD find backward links to rename them and update the original Document.</td>
</tr>
<tr>
<td>0x00000008</td>
<td>Return thicket folders or files.</td>
</tr>
<tr>
<td>0x00000010</td>
<td>Fix links within the same URL sub tree. Used when doing Link fixup after a directory has been renamed.</td>
</tr>
<tr>
<td>0x00000020</td>
<td>Allow renaming of Sites.</td>
</tr>
</tbody>
</table>
### Value | Description
--- | ---
0x00000040 | Allow the setting of the "CanBeParsed" Document flag when a file's extension changes.
0x00000080 | Allow update of the "CanHaveLinks" Document flag when a file's extension changes.
0x00000100 | Allow renaming of Sites and directory names.
0x00000200 | Allow move into the Forms Directory.
0x00000400 | Current User can view Draft Documents.
0x00000800 | Allow move operation on a Thicket with missing thicket supporting files.

**@PutFlags:** A 4-byte integer bit mask determining Document change options. This MAY have one or more flags set. The default value is 0, but it MUST NOT be NULL. The valid flags are:

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00000008</td>
<td>Keep the Document checked out.</td>
</tr>
<tr>
<td>0x00000020</td>
<td>Check in the Document.</td>
</tr>
<tr>
<td>0x00001000</td>
<td>Create a new displayed version of the Document, even if it is in a Short-Term Check Out.</td>
</tr>
<tr>
<td>0x00002000</td>
<td>Use client metadata for User, date and time for creation, last modification, and check in comments.</td>
</tr>
<tr>
<td>0x00010000</td>
<td>Publish the Document.</td>
</tr>
<tr>
<td>0x00020000</td>
<td>Overwrite the Document without updating its Displayed Version.</td>
</tr>
<tr>
<td>0x00100000</td>
<td>The Document is being added or updated as part of a system update. Do not update the last modification time and User.</td>
</tr>
<tr>
<td>0x00800000</td>
<td>Do not increment the internal version number for the Document. This flag SHOULD be set only if the User can tolerate having their changes overwritten by another User in the event of a conflict.</td>
</tr>
<tr>
<td>0x02000000</td>
<td>Keep the Document checked out to the User's local disk.</td>
</tr>
</tbody>
</table>

**@ReturnFlags:** A 4-byte integer bit mask determining the Result Sets returned from stored procedures called by proc_CopysUrl. This MAY have one or more flags set. The default value is 0, but it MUST NOT be NULL. The valid flags are in the following table, but the only one which has any effect in this case is 0x01.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00</td>
<td>Return no result set data.</td>
</tr>
<tr>
<td>0x01</td>
<td>Return Result Sets pertaining to renamed Documents</td>
</tr>
<tr>
<td>0x02</td>
<td>Return Result Sets pertaining to moved Documents with patched links.</td>
</tr>
</tbody>
</table>

**@AttachmentOp:** An integer value which governs the type of security checks that SHOULD be performed by the Stored Procedure on this Document's URL based on whether it appears to be an Attachment. The integer value MUST be listed in the following table:
<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Document is not an Attachment. Do not perform Attachment flag update.</td>
</tr>
<tr>
<td>1</td>
<td>Update the item’s flags only.</td>
</tr>
<tr>
<td>2</td>
<td>Update the item’s Version. In addition to performing an update on the Attachment flag, the Version number for the Attachment is to be updated along with the timestamp for when the Document was modified, the document level, and the Editor of the Document. This flag is set only if the User requesting the update has permissions to modify the List.</td>
</tr>
<tr>
<td>3</td>
<td>Update the item’s modification state. In addition to performing an update on the Attachment flag, the modification timestamp, the Level, and the associated Editor of the Attachment is to be updated.</td>
</tr>
</tbody>
</table>

@ParseDocsNow: Bit indicator that a Document needs further information gathering. Whenever a Document is moved into a new library, the metadata needs to be updated and @ParseDocsNow is set to 1. If the Document is left in the same library or moved within a library, @ParseDocsNow MUST be set to 0 or NULL.

@FailedUrl: If a Delete, Copy, or Move operation fails because of invalid parameters or permissions this will be filled in with the site-relative URL for the specific failed Document.

Return Code Values: An integer which MUST be listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful execution.</td>
</tr>
<tr>
<td>2</td>
<td>The specified destination was not found.</td>
</tr>
<tr>
<td>3</td>
<td>The specified Site or Subsite was not found.</td>
</tr>
<tr>
<td>5</td>
<td>User is not authorized to make this change</td>
</tr>
<tr>
<td>15</td>
<td>Attempt to rename an excluded directory type.</td>
</tr>
<tr>
<td>32</td>
<td>There was a sharing or lock violation.</td>
</tr>
<tr>
<td>33</td>
<td>Attempted to move directories that contain checked out files.</td>
</tr>
<tr>
<td>34</td>
<td>Attempt to rename a folder around List or move it out of List.</td>
</tr>
<tr>
<td>50</td>
<td>Attempt to rename a Site inside a List.</td>
</tr>
<tr>
<td>51</td>
<td>Attempted to rename a Forms folder.</td>
</tr>
<tr>
<td>53</td>
<td>There is an inconsistency between the specified and the expected value of @SubWebId. The only way this happens is if there are concurrent attempts made to change affected objects.</td>
</tr>
<tr>
<td>80</td>
<td>Invalid @PutFlags for a file operation.</td>
</tr>
<tr>
<td>87</td>
<td>There is an inconsistency between the expected number of Documents to be modified and the observed number which would be modified. The only way this happens is if there are concurrent attempts made to change affected objects.</td>
</tr>
<tr>
<td>130</td>
<td>Attempted to rename the thumbnail or image part of a thicket.</td>
</tr>
<tr>
<td>138</td>
<td>Attempted to copy Folders that span Lists.</td>
</tr>
<tr>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>144</td>
<td>Old and New URL object types are not the same.</td>
</tr>
<tr>
<td>161</td>
<td>Attempted to copy folders that span Sites.</td>
</tr>
<tr>
<td>190</td>
<td>Attempted to create a thicket.</td>
</tr>
<tr>
<td>206</td>
<td>Attempted to move folders that exceed file name range.</td>
</tr>
<tr>
<td>212</td>
<td>Write Lock Error when creating a file or directory.</td>
</tr>
<tr>
<td>214</td>
<td>Attempted to copy a thicket.</td>
</tr>
<tr>
<td>266</td>
<td>Specified Old and New URL are the same.</td>
</tr>
<tr>
<td>1150</td>
<td>Concurrency violation.</td>
</tr>
<tr>
<td>1359</td>
<td>Internal error occurred.</td>
</tr>
<tr>
<td>1816</td>
<td>Disk quota exceeded.</td>
</tr>
<tr>
<td>8389</td>
<td>At least one of the Lists could not be deleted.</td>
</tr>
</tbody>
</table>

**Result Sets:** The Stored Procedure MUST return multiple Result Sets. Some of the Result Sets are returned 0 or more times depending upon input parameters and type of URL to be copied. All Result Sets that are returned will be sent in the order listed.

### 3.1.4.13.1 List MetaData Result Set

This MUST be returned 0, 1, or 2 times. If the @ReturnFlags has the 0x01 bit set, a result set will be returned if the Old URL location has a containing List or if the New URL location has a containing List. The T-SQL syntax for the result set is as follows:

```sql
tp_ID          uniqueidentifier,
tp_Title       nvarchar(255),
tp_Modified    datetime,
tp_Created     datetime,
tp_LastDeleted datetime,
tp_Version     int,
tp_BaseType    int,
tp_FeatureId   uniqueidentifier,
tp_ServerTemplate int,
DirName        nvarchar(256),
LeafName       nvarchar(128),
DirName        nvarchar(256),
LeafName       nvarchar(128),
tp_ReadSecurity int,
tp_WriteSecurity int,
tp_Description ntext,
{tp_Fields}    ntext,
tp_Direction   int,
AnonymousPermMask bigint,
{tp_Flags}     bigint,
tp_ThumbnailSize int,
tp_WebImageWidth int,
tp_WebImageHeight int,
tp_WebImageUrl  nvarchar(255),
```
tp_ItemCount: The List Identifier of the List.

tp_Title: The title of this List for display in the user interface.

tp_Modified: A timestamp in Coordinated Universal Time (UTC) specifying when this List was last modified.

tp_Created: A timestamp in Coordinated Universal Time (UTC) specifying when this List was created.

tp_LastDeleted: A time stamp in Coordinated Universal Time (UTC) specifying when an item was last deleted from this List.

tp_Version: A counter incremented any time a change is made to the schema or other properties of this List, and is used for internal conflict detection.

tp_BaseType: This specifies the base type of the list. The value MUST be listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Generic List.</td>
</tr>
<tr>
<td>1</td>
<td>Document Library.</td>
</tr>
<tr>
<td>3</td>
<td>Discussion Board.</td>
</tr>
<tr>
<td>4</td>
<td>Survey List.</td>
</tr>
<tr>
<td>5</td>
<td>Issues List.</td>
</tr>
</tbody>
</table>

Copyright © 2012 Microsoft Corporation.

Release: July 16, 2012
**tp_FeatureId:** The feature identifier for the feature that defines the base schema of this List.

**tp_ServerTemplate:** The identifier for the template included in the feature definition or site definition that defines the base structure of this List.

**DirName:** The Directory Name of the location that contains this List.

**LeafName:** The Leaf Name of the location that contains this List.

**DirName:** The Directory Name of the default template Document in the List. This value MAY be NULL if a template Document is not defined for this List.

**LeafName:** The Leaf Name of the default template Document in the List. This value MAY be NULL if a template Document is not defined for this List.

**tp_ReadSecurity:** This signifies special restrictions that MAY be placed on List Item access. The value MUST be listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>No special restrictions.</td>
</tr>
<tr>
<td>2</td>
<td>Users SHOULD see only their own List Items. The front-end Web server MUST NOT display List Items to users without the ManageLists right unless the List Item was created by that User (for example tp_Author = @UserId).</td>
</tr>
</tbody>
</table>

**tp_WriteSecurity:** This signifies special restrictions that can be placed on List Item updates. The value MUST be listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>No special restrictions.</td>
</tr>
<tr>
<td>2</td>
<td>Users will see only their own List Items. The front-end Web server MUST NOT permit users without the ManageLists right to update a List Item unless the List Item was created by that User (find next tp_Author = @UserId).</td>
</tr>
<tr>
<td>4</td>
<td>Users will not update any List Items in this List. front-end Web server MUST NOT allow users without the ManageLists right to add or update List Items in this List.</td>
</tr>
</tbody>
</table>

**tp_Description:** The description of this List for display in the user interface.

**{tp_Fields}:** MUST be NULL if the Site or List has been flagged to cache all Schema data, otherwise contains an implementation-specific Version string followed by an XML representation of the field definitions. The field definitions include display and interaction options. See [MS-WSSCAML], section 2.4.1.12.

**tp_Direction:** An enumerated value specifying the direction of text flow for user interface elements presented by this List. The value MUST be listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No explicit direction is specified.</td>
</tr>
<tr>
<td>1</td>
<td>Text flow is left to right.</td>
</tr>
<tr>
<td>2</td>
<td>Text flow is right to left.</td>
</tr>
</tbody>
</table>
AnonymousPermMask: A flag mask that indicates the permissions granted to a User that is anonymous, or has no specific rights, on this List.

tpFlags: A list flags value describing this List.

tp_ThumbnailSize: The width, in pixels, to be used when creating thumbnail images of Documents within this List.

tp_WebImageWidth: The width, in pixels, to be used when creating Web Images of List Items within this List.

tp_WebImageHeight: The height, in pixels, to be used when creating Web Images of List Items within this List.

tp_ImageUrl: The URL of the image used to represent this List.

tp_ItemCount: The number of List Items that are stored within this List.

tp_Author: User Identifier of the List's creator.

tp_HasInternalFGP: Bit flag set to 1 if there have ever been List Items for this List that have had a unique Access Control List (ACL) applied.

tp_ScopeId: Security Scope Identifier for this List. This indicates the specific Access Control List (ACL) to use for calculating the permissions settings on this List.

Acl: The binary serialization of the Access Control List (ACL) for this List. This MAY be used for this List's permissions depending on the tp_ScopeId value of this List.

tp_EventSinkAssembly: The name of the assembly that contains the class definition of the event sink associated with this List.

tp_EventSinkClass: The name of the class definition for the Event Sink associated with this List.

tp_EventSinkData: Unicode string data specific to the implementation of the Event Sink associated with this List.

tp_EmailInsertsFolder: A URL fragment specifying the directory on the configured e-mail inserts server that is to be inspected for new e-mail messages to be processed for this List. If the list flags for this List do not have the value 0x000000000000000010000 set, this will be ignored.

tp_EmailInsertsLastSyncTime: A timestamp encoded as a Unicode string in yyyy-mm-dd hh:mi:ss.mmm format specifying the last time the location specified in the tp_EmailInsertsFolder column was inspected for new List Items. If the list flags value for this List does not have the value 0x000000000000000010000 set, this SHOULD be ignored.

tp_EmailAlias: The e-mail alias of the List. This alias is used to allow files to be sent directly to this List through an implementation-specific e-mail handling feature.

tp_WebFullUrl: The complete Store-Relative Form URL to the Site that contains this List.

tp_WebId: The Site Identifier of the Site that contains the List.

tp_WebTitle: The title, for display in the User interface, of the Site that contains this List.

tp_WebTemplate: The identifier of the site definition for the Site that contains this List.
**tp(WebLanguage):** The Language Code Identifier (LCID) of the display language of the Site that contains this List.

**tp(WebCollation):** The collation order for information in the Site that contains this List.

**tp(SendToLocation):** The title and URL for the "Send To Location" configured on this List. SendToLocation is an implementation specific feature that allows users to manually save copies of List Items and Documents to the remote location.

**{tp(MaxMajorVersionCount):** If the List has versioning enabled, this field contains the number of major versions that will be retained for this Document. All versions more than tp(MaxMajorVersionCount) removed from the current version of the Document are automatically removed at Version creation time. A value of 0 specifies that versions SHOULD NOT automatically be removed for this List.

**{tp(MaxMajorwithMinorVersionCount):** If the List has versioning enabled, this field contains the number of major versions that will have their associated minor versions retained for this Document. All versions more than tp(MaxMajorVersionCount) removed from the current version of the Document are automatically removed at Version creation time. A value of 0 specifies that versions SHOULD NOT automatically be removed for this List.

**tp(MaxRowOrdinal):** Specifies the maximum row ordinal used to store List Items for this List. This value indicates an implementation specific calculation for storage of List Items within Lists.

**tp(ListDataDirty):** Bit flag set to 1 if the List Items in this List require dependency update processing before their next access (for example, updating Document Link information by parsing each Document).

**tp(DefaultWorkflowId):** The workflow identifier corresponding to the workflow to be called if the Document is in a List which is a moderated object and the Document is submitted for approval as part of a check in. If the Document does not exist or is not contained in a List with a configured approval Workflow, this value MUST be NULL.

**tp(ContentTypes):** XML data specifying the Content Types registered for this List.

**tp(Subscription):** Bit flag set to 1 if an alert for changes to this List has been created in the past, signifying that additional processing needs to be performed.

### 3.1.4.13.2 NULL List Metadata Result Set

If copied URL has no lists, this MUST be returned 0, 1, or 2 times. It is returned twice if the @ReturnFlags does not have the 0x01 bit set; once for the Old URL Location, and once for the New URL Location. If the @ReturnFlags does have the 0x01 bit set, a NULL List Metadata Result Set will be returned if the Old URL location does not have a containing List, or if the New URL location does not have a containing File Result Set. This Result Set MUST be returned if the specified object is a File. It returns a log of the old file name and the new file names including the directory path information. The T-SQL syntax for the result set is as follows:

```
{OldUrlDirName}       nvarchar(256),
{OldUrlLeafName}      nvarchar(128),
{NewUrlDirName}       nvarchar(256),
{NewUrlLeafName}      nvarchar(128),
{Type}                int;
```

**{OldUrlDirName}:** Directory information as it exists before any transformation takes place.
\{\text{OldUrlLeafName}\}: The file name before any transformation takes place.

\{\text{NewUrlDirName}\}: Directory information as it exists after any transformation takes place.

\{\text{NewUrlLeafName}\}: The file name after any transformation takes place.

\{\text{Type}\}: Type of URL being transformed. This value MUST be 0.

### 3.1.4.13.3 Copied Directory Result Set

This Result Set MUST be returned if the specified object is a Directory. It returns a log of the old directory name and new directory name including directory path information. The T-SQL syntax for the result set is as follows:

\begin{verbatim}
OldDirName    nvarchar(256),
OldLeafName   nvarchar(128),
NewDirName    nvarchar(256),
NewLeafName   nvarchar(128),
Type          int;
\end{verbatim}

**OldDirName**: Directory information as it exists before any transformation takes place.

**OldLeafName**: The directory name before any transformation takes place.

**NewDirName**: Directory information as it exists after any transformation takes place.

**NewLeafName**: The directory name after any transformation takes place.

**Type**: Type of URL being transformed. This value MUST be 1.

### 3.1.4.14 proc_CreateList

The proc_CreateList Stored Procedure is called to create a new entry in the Content Database for the specified List and to return its Metadata and Full URL. The Stored Procedure is defined using T-SQL syntax, as follows:

\begin{verbatim}
PROCEDURE proc_CreateList(
    @SiteId                        uniqueidentifier,
    @WebId                         uniqueidentifier,
    @ListId                        uniqueidentifier,
    @DirName                       nvarchar(256),
    @FolderNameBase                nvarchar(50),
    @bAlternateUrlOnCollision      bit,
    @Title                         nvarchar(255),
    @Author                        int,
    @BaseType                      int,
    @bCreateAttachmentsSubFolder   bit,
    @FeatureId                     uniqueidentifier,
    @ServerTemplate                int,
    @DocLibTemplate                uniqueidentifier,
    @ImageUrl                      nvarchar(255),
    @ReadSecurity                  int,
    @WriteSecurity                 int,
    @Description                   ntext,
    @MajorVersionCount             int,
    @MinorVersionCount             int,
\)
\end{verbatim}
@Fields
text,
@Direction
int,
@Flags
bigint,
@ThumbnailSize
int,
@WebImageWidth
int,
@WebImageHeight
int,
@bParentFolderChecked
bit,
@OnRestore
bit,
@EventSinkAssembly
nvarchar(255),
@EventSinkClass
nvarchar(255),
@EventSinkData
nvarchar(255),
@ContentTypes
ntext,
@RootFolderId
uniqueidentifier = NULL,
@FolderFullUrlRet
nvarchar(256) = NULL OUTPUT,
@TimeCreated
datetime = NULL
}

@SiteId: The Site Collection Identifier of the Site Collection which contains the specified Site.

@WebId: The Site Identifier of the Site which contains the specified List.

@ListId: The List Identifier of the List that is being created.

@DirName: The Directory Name of the List location.

@FolderNameBase: The Home directory base name used to generate a unique directory name.

@bAlternateUrlOnCollision: A bit that specifies whether or not to generate a unique alternate location for the Home directory in the case of a name collision.

@Title: The Title of the specified List.

@Version: The initial list version to start with.

@Author: The User Identifier of the list author.

@BaseType: This specifies the list base type of this List. The value MUST be in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Generic List</td>
</tr>
<tr>
<td>1</td>
<td>Document Library</td>
</tr>
<tr>
<td>3</td>
<td>Discussion Board</td>
</tr>
<tr>
<td>4</td>
<td>Survey List</td>
</tr>
<tr>
<td>5</td>
<td>Issues List</td>
</tr>
</tbody>
</table>

@bCreateAttachmentsSubFolder: A bit specifying whether or not to create a subfolder for List attachments.

@FeatureId: The Feature Identifier of the Feature associated with the List.

@ServerTemplate: The integer value of the list template that defines the base structure of this List.
@DocLibTemplate: The List Template of the Document Library.

@ImageUrl: Contains the server-relative URL that points to an image associated with the List.

@ReadSecurity: A value identifying the security policy for read access on List Items. If set this value is set to 1, Users with read permissions MAY read all List Items. Otherwise, if this value is set to 2, then Users with read permissions can only read their own List Items.

@WriteSecurity: A value identifying the security policy for write access on List Items. The value MUST be in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Users with write permissions have write access to all List Items</td>
</tr>
<tr>
<td>2</td>
<td>Users with write permissions have write access to their own List Items only</td>
</tr>
<tr>
<td>4</td>
<td>Users have no write access to any List Items</td>
</tr>
</tbody>
</table>

@Description: The text describing the List.

@MajorVersionCount: Sets the maximum number of major versions to be retained by the List.

@MinorVersionCount: Sets the maximum number of Minor Versions to be retained by the List.

@Fields: A version string followed by the XSD representation of the Field Definitions. The field definitions include display and interaction options. See [MS-WSSCAML] section 2.4.1.12.

@Direction: An enumerated value specifying the direction of text flow for user interface elements presented by this List. The value MUST be in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No explicit direction is specified.</td>
</tr>
<tr>
<td>1</td>
<td>Text flow SHOULD be left to right.</td>
</tr>
<tr>
<td>2</td>
<td>Text flow SHOULD be right to left.</td>
</tr>
</tbody>
</table>

@Flags: A bit array for setting List functionality. This parameter MUST not be NULL. Valid values are contained in the table defined in [MS-WSSFO] section 2.2.2.5.

@ThumbnailSize: An integer used by Lists to determine the rendering size of an image thumbnail. If this parameter NULL, then it MUST be ignored.

@WebImageWidth: An integer used by Lists to determine the rendering width of an image. If this parameter NULL, then it MUST be ignored.

@WebImageHeight: An integer used by Lists to determine the rendering height of an image. If this parameter NULL, then it MUST be ignored.

@bParentFolderChecked: A bit that specifies whether or not the list parent folder has been checked for existence.

@OnRestore: A Boolean value indicating that this List is undergoing a backup restore operation. For more information, see proc_AddListItem in [MS-WSSFO].
@EventSinkAssembly: An assembly name for an event handler of an Event Sink for the List. If this parameter NULL, then it MUST be ignored.

@EventSinkClass: An assembly class identifier for an event handler of an Event Sink for the List. If this parameter NULL, then it MUST be ignored.

@EventSinkData: Event Sink data for an event handler of an Event Sink for the List. If this parameter NULL, then it MUST be ignored.

@ContentTypes: The XSD representation of the Content Types available to this List.

@RootFolderId: The root folder identifier for this List.

@FolderFullUrlRet: The Full URL of the List.

@TimeCreated: The date and time that the List was created.

Return Values: The Stored Procedure returns an integer Return Code which MUST be in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful execution.</td>
</tr>
<tr>
<td>3</td>
<td>The system cannot find the path specified.</td>
</tr>
<tr>
<td>5</td>
<td>Access is denied.</td>
</tr>
<tr>
<td>80</td>
<td>The List with the specified title already exists.</td>
</tr>
<tr>
<td>87</td>
<td>Invalid parameter specified (that is, @SiteId, @WebId, @FolderNameBase).</td>
</tr>
<tr>
<td>212</td>
<td>The path segment is locked and cannot be reallocated.</td>
</tr>
<tr>
<td>1836</td>
<td>Not enough quota is available to process this command.</td>
</tr>
</tbody>
</table>

Result Sets: The Stored Procedure MUST return 1 Result Set when the Return Code is 0, otherwise it MUST not return any Result Sets.

3.1.4.14.1 List Metadata Result Set

Returns the list metadata for the newly created List. This Result Set will be returned when input parameter @BaseType has a value of 1. This Result Set is defined in the [MS-WSSFO], section 2.2.5.12.

3.1.4.14.2 Id and Full URL Result Set

Returns the List Identifier and Full URL of the new List. This Result Set will be returned when input parameter @BaseType has a value different than 1. The Result Set is defined using T-SQL syntax, as follows:

```sql
{ListId} uniqueidentifier,
{FolderFullUrl} nvarchar(256);
```

{ListId}: Contains the List Identifier of the List that has been created.
3.1.4.15 proc_CreateSite

The proc_CreateSite Stored Procedure is called to create a new Site Collection with the specified Metadata. The Stored Procedure is defined using T-SQL syntax, as follows:

```sql
PROCEDURE proc_CreateSite(
    @SiteId                   uniqueidentifier,
    @DirName                  nvarchar(256),
    @LeafName                 nvarchar(128),
    @RootWebUrl               nvarchar(256),
    @Language                 int,
    @Collation                smallint,
    @CalendarType             smallint,
    @Time24                   bit,
    @OwnerSID                 varbinary(512),
    @OwnerLogin               nvarchar(255),
    @OwnerName                nvarchar(255),
    @OwnerEmail               nvarchar(255),
    @SecondaryContactSID     varbinary(512),
    @SecondaryContactLogin   nvarchar(255),
    @SecondaryContactName    nvarchar(255),
    @SecondaryContactEmail   nvarchar(255),
    @AdminsName               nvarchar(255),
    @AdminsDescription        nvarchar(512),
    @AdminsPermMask           bigint,
    @AuthorsName              nvarchar(255),
    @AuthorsDescription       nvarchar(512),
    @AuthorsPermMask          bigint,
    @ContributorsName         nvarchar(255),
    @ContributorsDescription  nvarchar(512),
    @ContributorsPermMask     bigint,
    @BrowsersName             nvarchar(255),
    @BrowsersDescription      nvarchar(512),
    @BrowsersPermMask         bigint,
    @GuestsName               nvarchar(255),
    @GuestsDescription        nvarchar(512),
    @GuestsPermMask           bigint,
    @SiteHashKey              binary(16),
    @HostHeader               nvarchar(260)
);
```

@SiteId: The Site Collection Identifier of the Site Collection to be created.

@DirName: The directory name of the specified location. This parameter SHOULD be empty in the case of the host header Site Collection.

@LeafName: The Leaf Name of a Site Collection. This parameter SHOULD be empty in the case of the Host Header Site Collection.

@RootWebUrl: The virtual path relative to the top-level site. This parameter SHOULD be empty in the case of the Host Header Site Collection.

@Language: The Language Code Identifier (LCID) for the new Site Collection.

@Collation: The Identifier that specifies the Collation Order.
@CalendarType: The Identifier that specifies the calendar type that is being used.

@Time24: A Bit Flag which specifies whether a 24-hour time format SHOULD be used when displaying time values. If this parameter is set to 1, the 24-hour time format SHOULD be used; otherwise, the 12-hour time format SHOULD be used.

@OwnerSID: The SystemID of the owner of the Site Collection.

@OwnerLogin: The Login Name of the owner of the Site Collection.

@OwnerName: The display name of the owner of the Site Collection.

@OwnerEmail: The e-mail address of the owner of the Site Collection.

@SecondaryContactSID: The SystemID of the secondary contact of the Site Collection.

@SecondaryContactLogin: The Login Name of the secondary contact of the Site Collection. This parameter is ignored when @SecondaryContactSID is NULL or equal to @OwnerSID.

@SecondaryContactName: The Display Name of the secondary contact of the Site Collection. This parameter is ignored when @SecondaryContactSID is NULL or equal to @OwnerSID.

@SecondaryContactEmail: The e-mail address of the secondary contact of the Site Collection. This parameter is ignored when @SecondaryContactSID is NULL or equal to @OwnerSID.

@AdminsName: The Display Name of the Site Group for Administrators.

@AdminsDescription: The description for Site Group for Administrators.

@AdminsPermMask: An access mask containing the rights that SHOULD be granted to the Site Group for Administrators in the Site Collection.

@AuthorsName: The Display Name of the Site Group for site authors.

@AuthorsDescription: The description of the Site Group role for site authors.

@AuthorsPermMask: An access mask containing the rights that SHOULD be granted to the Site Group for site authors.

@ContributorsName: The Display Name of the Site Group for site contributors.

@ContributorsDescription: The description of the Site Group Role for site contributors.

@ContributorsPermMask: An access mask containing the rights that SHOULD be granted to the Site Group for Site Contributors.

@BrowsersName: The Display Name of the Site Group for Site Browsers.

@BrowsersDescription: The description of the Site Group Role for Site Browsers.

@BrowsersPermMask: An access mask containing the list of rights that SHOULD be granted to the Site Group for Site Browsers.

@GuestsName: The Display Name of the Site Group for Guests.

@GuestsDescription: The description of the Site Group Role for Guests.

@GuestsPermMask: An access mask containing the rights that SHOULD be granted to the Site Group for Guests.
@SiteHashKey: The Hash Key of this Site Collection. It is a random set of 16 bytes which are used to generate the form digest validation for this Site Collection. This parameter MAY be NULL.

@HostHeader: The Host Header of this Site Collection. This parameter MAY be NULL when it is not a Host Header.

Return Code Values: An integer which MUST be listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful execution.</td>
</tr>
<tr>
<td>80</td>
<td>The Site Collection Identifier already exists in the database.</td>
</tr>
</tbody>
</table>

Result Sets: proc_CreateSite MUST return up to 7 Result Sets. Some of the Result Sets are returned conditionally. Result Sets from this Stored Procedure SHOULD be ignored by the caller. All Result Sets returned will be sent in the following order:

3.1.4.15.1 Site Owner Audit Mask Result Set

The Site Owner Audit Mask Result Set returns the information about the audit flags associated with the Site owner. The Site Audit Mask Result Set MUST return a single row. It is defined in [MS-WSSEUX], section 3.1.4.50.1.

3.1.4.15.2 Site Secondary Contact Audit Mask Result Set

The Secondary Contact Site Audit Mask Result Set returns the information about the Audit Flags associated with the Site secondary contact. This Result Set MUST not be returned if such a contact has not been specified. For information about the definition, refer to [MS-WSSEUX], section 3.1.4.50.1.

3.1.4.15.3 Site Administrator Audit Mask Result Set

The Site Administrator Audit Mask Result Set returns the information about the Audit Flags associated with the Site administrator. This Result Set MUST not be returned if such a contact has not been specified. For information about the definition, refer to [MS-WSSEUX], section 3.1.4.50.1.

3.1.4.15.4 Site Author Audit Mask Result Set

The Site Author Audit Mask result set returns the information about the audit flags associated with the site author. This result set MUST not be returned if such a contact has not been specified. For information about the definition, see [MS-WSSEUX] section 3.1.4.50.1.

3.1.4.15.5 Site Contributor Audit Mask Result Set

The Site Contributor Audit Mask Result Set returns the information about the Audit Flags associated with the Site contributor. This Result Set MUST not be returned if such a contact has not been specified. For information about the definition, refer to [MS-WSSEUX], section 3.1.4.50.1.

3.1.4.15.6 Site Browser Audit Mask Result Set

The Site Browser Audit Mask Result Set returns the information about the Audit Flags associated with the Site browser. This Result Set MUST not be returned if such a contact has not been specified. For information about the definition, refer to [MS-WSSEUX], section 3.1.4.50.1.
3.1.4.15.7 Site Guest Audit Mask Result Set

The Site Guest Audit Mask Result Set returns the information about the Audit Flags associated with the Site guest. This Result Set MUST not be returned if such a contact has not been specified. For information about the definition, refer to [MS-WSSEUX], section 3.1.4.50.1.

3.1.4.16 proc_CreateView

The proc_CreateView Stored Procedure is called to create a new View for the specified List. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_CreateView(
    @SiteId uniqueidentifier,
    @WebId uniqueidentifier,
    @ViewId uniqueidentifier,
    @Level tinyint,
    @ListId uniqueidentifier,
    @Type tinyint,
    @Flags int,
    @BaseViewID tinyint,
    @DisplayName nvarchar(255),
    @ContentTypeId varbinary(512),
    @DocId uniqueidentifier,
    @WebPartTypeID uniqueidentifier,
    @ZoneId nvarchar(64),
    @PartOrder int,
    @ViewOrder int,
    @View ntext,
    @source ntext = NULL
);
```

@SiteId: The Site Collection Identifier of the Site Collection to contain the specified View for the specified List.

@WebId: The Site Identifier of the Site which contains the specified newly created View for the specified List.

@ViewId: A view identifier for the newly created View for the specified List.

@Level: The Publishing Level for the newly created View. This parameter MUST be 1.

@ListId: The List Identifier of the List for which the new View is being created.

@Type: The page type for the newly created View.

@Flags: The view flags for the newly created View. Refer to [MS-WSSFO], section 2.2.2.11 for valid values.

@BaseViewID: The base view identifier for the newly created View. MUST be unique per View defined for the specified List.

@DisplayName: The Unicode string which represents a user-friendly name for the newly created View.

@ContentTypeId: The Content Type Identifier for the newly created View.
@DocId: The Document Identifier of the Document that will contain the newly created View. This parameter MUST NOT be NULL.

@WebPartTypeId: The Web Part type identifier of the Web Part for the newly created View. MUST NOT be NULL.

@ZoneId: The name of the Web Part zone that the specified Web Part will be positioned in the specified Document.

@PartOrder: A zero-based index integer specifying the order that the Web Part will be displayed in the specified Document. Each Web Part within the specified Web Part Zone will be displayed in increasing order. MUST NOT be NULL.

@ViewOrder: A zero-based index integer specifying the order that the newly created View will be shown in relation to existing Views for the specified List. When displaying Views for a given List, the Views MUST be listed in increasing order as specified by @ViewOrder. If this parameter is NULL, then the newly created View MUST be assigned an order greater than the order of all other Views for the specified List.

@View: A query expressed in Collaborative Application Markup Language (CAML) used when processing this view. See [MS-WSSCAML] for more information about the Collaborative Application Markup Language (CAML).

@source: The Web Part property or properties of the Web Part in either WPV2:WebPart format, see [MS-WPPS], section 2.2.4.2, or HTML format.

Return Code Values: An integer which MUST be listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>The Stored Procedure execution has finished. The Stored Procedure MAY have failed.</td>
</tr>
<tr>
<td>1816</td>
<td>The View for the specified List could not be created because creating the new View would have exceeded the Site Collection Quota.</td>
</tr>
<tr>
<td>212</td>
<td>The View for the specified List could not be created because the Site Collection has its WRITELOCK Site Collection Flag bit set.</td>
</tr>
<tr>
<td>3</td>
<td>The Document specified by the @DocId Stored Procedure parameter exists in a Site whose Site Identifier is not the same as the Site Identifier specified by the @WebId Stored Procedure parameter.</td>
</tr>
<tr>
<td>1</td>
<td>An error occurred and the View was not created.</td>
</tr>
</tbody>
</table>

Result Sets: MUST NOT return any Result Sets.

3.1.4.17 proc_CreateWeb

The proc_CreateWeb Stored Procedure is called to create a new site. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_CreateWeb(
    @WebSiteId uniqueidentifier,
    @WebId uniqueidentifier,
    @WebDirName nvarchar(256),
    @WebLeafName nvarchar(128),
    @WebFullUrl nvarchar(260),
)
```
@ProductVersion smallint,
@TemplateVersion smallint,
@Language int,
@Collation smallint,
@CalendarType smallint,
@AuthorID int,
@Time24 bit,
@ConvertIfThere bit,
@UniqueWeb bit,
@NewWebId uniqueidentifier = NULL,
@DocId uniqueidentifier = NULL
);

@WebSiteId: The Site Collection Identifier of the Site Collection that will contain the new Site specified by the @WebId Stored Procedure parameter.

@WebId: The Site Identifier of an existing Site that will become the Parent Site for the Site to be created.

@WebDirName: The Directory Name for the new Site.

@WebLeafName: The Leaf Name for the new Site.

@WebFullUrl: This Stored Procedure parameter is ignored.

@ProductVersion: MUST be 3.

@TemplateVersion: The site definition version of the site definition for the new Site.

@Language: The Language Code Identifier (LCID) for the new Site.

@Collation: The Collation Order for the new Site.

@CalendarType: The Calendar Type for the new Site.

@AuthorID: The User Identifier of the User that is creating the new Site.

@Time24: MUST be one of the following:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>If the new Site is to display time using the 12-hour clock notation.</td>
</tr>
<tr>
<td>1</td>
<td>If the new Site is to display time using the 24-hour clock notation.</td>
</tr>
</tbody>
</table>

@ConvertIfThere: If this Stored Procedure parameter is set to 1 and an existing Folder is located at the URL specified by combining the @WebDirName and @WebLeafName Stored Procedure parameters, then the existing Folder is converted to a Site. If this Stored Procedure parameter is set to 0 and an existing Folder is located at the URL specified by combining the @WebDirName and @WebLeafName Stored Procedure parameters, then the existing Folder is not converted to a Site.

@UniqueWeb: When the new Site is created, the Role Assignments from the Site Collection are applied to (or inherited by) the Site. Therefore, the Security Scope of the various Role Assignments applied to the Site Collection also applies to any Site contained within the Site Collection. The Site Collection and its Site are said to have the same Security Scope. However, if this Stored Procedure parameter is set to 1, the new Site will still inherit the Role Assignments of its Site Collection, but the Site will no longer have the same Security Scope as its Site Collection.
@NewWebId: This Stored Procedure parameter MUST be NULL.

@DocId: This Stored Procedure parameter MUST be NULL.

Return Code Values: An integer which MUST be listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful execution.</td>
</tr>
</tbody>
</table>
| 3     | This Return Code MUST be returned if any of the following are true:  
- The Site Collection specified by the @WebSiteId Stored Procedure parameter does not exist.  
- The Site specified by the @WebId Stored Procedure parameter exists, but is not contained within the Site Collection specified by the @WebSiteId Stored Procedure.  
- The Site specified by the @WebId Stored Procedure parameter does not exist. |
| 80    | This Return Code MUST be returned if any of the following are true:  
- A Document or Site already exists at the URL specified by combining the @WebDirName and @WebLeafName Stored Procedure parameters.  
- A failure occurred and the new Site MAY NOT have been created. |
| 85    | If the URL specified by combining the @WebDirName and @WebLeafName Stored Procedure parameters is an existing Folder and the @ConvertIfThere Stored Procedure parameter is 0, the new Site is created but will not be operational because the Folder could not be converted to a functional Site. |
| 161   | The Site creation operation MUST fail with this Return Code if all of the following are true:  
1. The URL specified by combining the @WebDirName and @WebLeafName Stored Procedure parameters is an existing Folder.  
2. The @ConvertIfThere Stored Procedure parameter is 1.  
3. The existing Folder or any of its child Folders is already contained within another Site. |
| 138   | The Site creation operation MUST fail with this Return Code if all of the following are true:  
1. The URL specified by combining the @WebDirName and @WebLeafName Stored Procedure parameters is an existing Folder.  
2. The @ConvertIfThere Stored Procedure parameter is 1.  
3. The existing Folder or any of its child Folders is already contained within a List. |
| 33    | The Site creation operation MUST fail with this Return Code if all of the following are true:  
1. The URL specified by combining the @WebDirName and @WebLeafName Stored Procedure parameters is an existing Folder.  
2. The @ConvertIfThere Stored Procedure parameter is 1.  
3. The existing Folder or any of its child Folders contains a Document that is Checked |
206
The Site creation operation MUST fail with this Return Code if all of the following are true:
1. The URL specified by combining the @WebDirName and @WebLeafName Stored Procedure parameters is an existing Folder.
2. The @ConvertIfThere Stored Procedure parameter is 1.
3. The existing Folder or any of its child Folders exceeds the maximum Directory Name of 256 Unicode characters.

212
The Site could not be created because the Site Collection has its WRITELOCK Site Collection Flag bit set.

1816
The Site could not be created because creating the new Site would have exceeded the Site Collection Quota.

Result Sets: proc_CreateWeb MUST return the Audit Flags Result Set if the @UniqueWeb parameter was set to 1 on input. If the @UniqueWeb parameter was set to something other than 1 on input, proc_CreateWeb MUST NOT return the Audit Flags Result Set.

3.1.4.17.1 Audit Flags Result Set

The proc_CreateWeb stored procedure returns the Audit Flags for the newly created site. The Audit Flags Result Set MUST return 1 row. The Audit Flags Result Set is defined using T-SQL syntax, as follows:

```sql
{WebId} uniqueidentifier,
{WebAuditFlags} int,
{WebInheritAuditFlags} int,
{SiteCollectionAuditFlags} int;
```

{WebId}: The site identifier for the newly created site.

{WebAuditFlags}: The Audit Flags for the newly created site.

{WebInheritAuditFlags}: The Audit Flags that are inherited from the newly created site's parent site.

{SiteCollectionAuditFlags}: The Audit Flags of the site collection that contains the newly created site.

3.1.4.18 proc_DeleteAllItemVersions

The proc_DeleteAllItemVersions Stored Procedure is called to delete all Historical Versions of a given List Item. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_DeleteAllItemVersions(
@SiteId uniqueidentifier,
@WebId uniqueidentifier,
@ListId uniqueidentifier,
@ItemId int,
```
@UserId: The user identifier for the current user.

@UseNvarchar1ItemName: This bit flag specifies whether to use the content of the nvarchar1 column for the Display Name of the List Item for purposes of tracking in the Recycle Bin.

@DeleteOp: A parameter specifying the delete options. The value MUST be listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>The deleted Document versions MUST NOT be placed in the Recycle Bin (non-recoverable delete).</td>
</tr>
<tr>
<td>4</td>
<td>The deleted Document versions MUST be placed in the Recycle Bin (recoverable delete).</td>
</tr>
</tbody>
</table>

Return Code Values: An integer which MUST be listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful execution.</td>
</tr>
<tr>
<td>2</td>
<td>The List Item specified cannot be found in the Site Collection.</td>
</tr>
</tbody>
</table>

Result Sets: MUST NOT return any Result Sets.

3.1.4.19 proc_DeleteAttachment

The proc_DeleteAttachment Stored Procedure is called to delete an Attachment from a List Item. The T-SQL syntax for the Stored Procedure is as follows:

```
PROCEDURE proc_DeleteAttachment(
    @SiteId uniqueidentifier,
    @WebId uniqueidentifier,
    @FolderUrl nvarchar(256),
    @RowID uniqueidentifier,
    @UserId int,
    @DeleteOp int
);
```

@SiteId: The Site Collection Identifier of the Site Collection which contains the specified Site.

@WebId: The Site Identifier of the Site which contains the specified List Item Attachment to be deleted.
@FolderUrl: The Store-Relative Form URL path of the List Item containing the attachment to be deleted.

@RowID: The Attachment identifier for the Attachment to be deleted.

@UserId: The identifier for the user performing the delete operation.

@DeleteOp: The value that specifies the type of delete operation to perform. This parameter MUST specify a valid value from the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>Delete attachment.</td>
</tr>
<tr>
<td>4</td>
<td>Send attachment to the Recycle Bin. If Recycle Bin is not available, do not delete the attachment.</td>
</tr>
<tr>
<td>5</td>
<td>Send attachment to the Recycle Bin. If Recycle Bin is not available, delete the attachment.</td>
</tr>
</tbody>
</table>

Return Code Values: An integer which MUST be 0.

Result Sets: MUST NOT return any Result Sets.

3.1.4.20 proc_DeleteAttachmentsFolder

The proc_DeleteAttachmentsFolder Stored Procedure is called to disable attachments on a List and remove all existing attachments. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_DeleteAttachmentsFolder(
   @SiteId uniqueidentifier,
   @WebId uniqueidentifier,
   @ListId uniqueidentifier,
   @UserId int
);
```

@SiteId: The Site Collection Identifier of the Site Collection which contains the specified Site.

@WebId: The Site Identifier of the Site which contains the specified List.

@ListID: The List Identifier of the List which contains the attachments folder to be deleted.

@UserId: The identifier for the user performing the delete operation.

Return Code Values: An integer which MUST be listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful execution.</td>
</tr>
<tr>
<td>5</td>
<td>User is not authorized to make this change.</td>
</tr>
<tr>
<td>33</td>
<td>Cannot delete Attachments folder containing Checked Out or locked files.</td>
</tr>
<tr>
<td>50</td>
<td>Cannot delete Attachments folder.</td>
</tr>
<tr>
<td>1150</td>
<td>Concurrency violation or unknown error occurred.</td>
</tr>
</tbody>
</table>

Result Sets: MUST NOT return any Result Sets.
3.1.4.21 proc_DeleteCategory

The proc_DeleteCategory Stored Procedure is called to disassociate a Category with a Site. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_DeleteCategory(
    @SiteId       uniqueidentifier,
    @WebDirName   nvarchar(256),
    @WebLeafName  nvarchar(128),
    @Category     nvarchar(128)
);
```

@SiteId: The Site Collection Identifier of the Site Collection which contains the specified List.

@WebDirName: The Directory Name of the Site to which the Category is to be disassociated.

@WebLeafName: The Leaf Name of the Site to which the Category is to be disassociated.

@Category: Category to delete from the Site.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST NOT return any Result Sets.

3.1.4.22 proc_DeleteChanges

The proc_DeleteChanges Stored Procedure is called to delete the Events in the Change Log older than the specified number of days. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_DeleteChanges(
    @days     int
);
```

@days: An integer that specifies a number of days. Events in the Change Log older than the specified number of @days will be Deleted.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST NOT return any Result Sets.

3.1.4.23 proc_DeleteEventLog

The proc_DeleteEventLog Stored Procedure is called to delete an event log, information stored in Back-End Database Server that is used to generate Alerts. Events created before the specified Coordinated Universal Time (UTC) time will be Deleted. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_DeleteEventLog(
    @SiteId       uniqueidentifier,
    @EventTime    datetime
);
```

@SiteId: The Site Collection Identifier of the Site Collection which contains the event log which will be deleted.
@EventTime: A Coordinated Universal Time (UTC) time. Events created before this time will be deleted.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST NOT return any Result Sets.

3.1.4.24 proc_DeleteItemVersion

The proc_DeleteItemVersion Stored Procedure is called to delete a Historical Version of a List Item. The T-SQL syntax for the Stored Procedure is as follows:

```t-sql
PROCEDURE proc_DeleteItemVersion(  
    @SiteId                  uniqueidentifier,  
    @WebId                   uniqueidentifier,  
    @ListId                  uniqueidentifier,  
    @ItemId                  int,  
    @ItemVersion             int,  
    @UserId                  int,  
    @UseNvarchar1ItemName    bit = 1,  
    @DeleteOp                int = 3  
);  
```

@SiteId: The Site Collection Identifier of the Site Collection which contains the specified List Item.

@WebId: The Site Identifier for the Site containing the specified List Item. This parameter MUST correspond to a valid Site, and MUST NOT be NULL.

@ListId: The List Identifier of the List which contains the specified List Item.

@ItemId: The item identifier for the specified List Item. This parameter MUST correspond to a valid List Item, and MUST NOT be NULL.

@ItemVersion: The User Interface (UI) Version for the specified List Item. This parameter MUST correspond to a valid User Interface (UI) Version, and MUST NOT be NULL.

@UserId: The user identifier for the current user.

@UseNvarchar1ItemName: This bit flag specifies whether to use the content of the nvarchar1 column for the Display Name of the List Item for purposes of tracking in the Recycle Bin.

@DeleteOp: A parameter specifying the delete options. The value MUST be listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>The deleted Document versions MUST NOT be placed in the Recycle Bin (non-recoverable delete).</td>
</tr>
<tr>
<td>4</td>
<td>The deleted Document versions MUST be placed in the Recycle Bin (recoverable delete).</td>
</tr>
</tbody>
</table>

Return Code Values: An integer which MUST be listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful execution.</td>
</tr>
<tr>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>2</td>
<td>The List Item specified cannot be found in the Site Collection.</td>
</tr>
</tbody>
</table>

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

### 3.1.4.25 proc_DeleteSite

The proc_DeleteSite Stored Procedure is called to delete a Site Collection. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_DeleteSite(
    @SiteId    uniqueidentifier
);
```

[@SiteId]: The Site Collection Identifier of the Site Collection to be deleted.

**Return Code Values**: An integer which MUST be 0.

**Result Sets**: MUST return two Result Sets as follows:

#### 3.1.4.25.1 Site Collection Flags Result Set

The Site Collection Flags Result Set returns information about the Site Collection that has been deleted. The Site Collection Flags Result Set MUST return one row if the Site Collection specified by the @SiteId Stored Procedure parameter existed in the Content Database and was deleted. If the Site Collection specified by the @SiteId Stored Procedure parameter did not exist in the Content Database then the Site Collection Flags Result Set MUST NOT return any rows. The T-SQL syntax for the result set is as follows:

```sql
BitFlags    int;
```

**BitFlags**: Contains the Site Collection Flags of the Site Collection specified by the @SiteId Stored Procedure parameter.

#### 3.1.4.25.2 Distribution List E-mail Address Result Set

Distribution List E-mail Address Result Set returns information about the Site Collection that has been deleted. If the Site Collection specified by the @SiteId Stored Procedure parameter existed in the Content Database and was deleted, then the Distribution List E-mail Address Result Set MUST return one row for each of the Site Collection's security groups that contains a distribution list e-mail address If, however, the deleted Site Collection's security groups do not contain any distribution list e-mail addresses, then the Distribution List E-mail address Result Set MUST NOT return any rows. If the Site Collection specified by the @SiteId Stored Procedure parameter did not exist in the Content Database then the Distribution List E-mail Address Result Set MUST NOT return any rows. The T-SQL syntax for the result set is as follows:

```sql
DLAlias    nvarchar(128);
```

**DLAlias**: Contains a Distribution list e-mail address for the deleted Site Collection specified by the @SiteId Stored Procedure parameter.
3.1.4.26  proc_DeleteSiteAsync

The proc_DeleteSiteAsync Stored Procedure is called to delete a site collection. Unlike proc_DeleteSite, the server will not perform the deletion right away. Instead, the server will store the identifier of the site, and assign a deletion identifier, as specified in section 3.1.4.27, to it. The implementer of this protocol MUST implement an application that queries the server about the list of sites waiting to be deleted and call proc_DeleteSiteInternalAsync to delete them. T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_DeleteSiteAsync(
    @SiteId    uniqueidentifier
);
```

@SiteId: The Site Collection Identifier of the Site Collection to be deleted.

Return Code Values: An integer that MUST be zero.

Result Sets: None.

3.1.4.27  proc_GetSiteDeletionBatch

The proc_GetSiteDeletionBatch Stored Procedure is called to get a list of site collections which are waiting to be deleted. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE dbo.proc_GetSiteDeletionBatch(
    @DeletionId bigint
)
```

@DeletionId: An 8-byte integer number generated by the back-end database server when the client makes a proc_DeleteSiteAsync call. The server MUST generate this number in a monotonically increasing order. The server stores this number and associates it to the site identifier parameter from the proc_DeleteSiteAsync call. Several methods in this protocol refer to this number. If this parameter is NULL, the server returns up to 1,000 site collections which are waiting to be deleted. If this parameter is NOT NULL, the server returns up to 1,000 site collection whose deletion identifier is bigger than this parameter.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST return one two Result Sets as follows:

3.1.4.27.1 Site Collection Deletion Batch Result Set

The Site Collection Deletion Batch Result Set returns information about up to 1,000 site collections which are waiting to be deleted. The T-SQL syntax for the result set is as follows:

```sql
Id    int;
SiteId    uniqueidentifier;
```

Id: The Deletion Identifier, as specified in section 3.1.4.27, of the site collection.

SiteId: The Identifier of the site to be deleted.
3.1.4.28 proc_DeleteSiteInternalAsync

The proc_DeleteSiteInternalAsync Stored Procedure is called to delete a site collection. When the server finishes deleting the site collection, it will remove the site collection from the list which contains site collections which are waiting to be deleted.

```sql
PROCEDURE proc_DeleteSiteInternalAsync(
    @SiteId    uniqueidentifier,
    @DeletionId bigint
);
```

@SiteId: The Identifier of the site to be deleted

@DeletionId: The Deletion Identifier, as specified in section 3.1.4.27, of the site collection.

Return Code Values: An integer which MUST be 0.

Result Sets: None.

3.1.4.29 proc_DeleteView

The proc_DeleteView Stored Procedure is called to delete a View from the specified List. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_DeleteView(
    @SiteId                    uniqueidentifier,
    @ListId                    uniqueidentifier,
    @ViewId                    uniqueidentifier,
    @CanManagePersonalViews    bit,
    @CanManageLists            bit,
    @UserId                    int
);
```

@SiteId: The Site Collection Identifier of the Site Collection which contains the specified View for the specified List.

@ListId: The List Identifier of the List that contains the specified View.

@ViewId: The View Identifier of the View that is to be deleted.

@CanManagePersonalViews: If the specified View to be deleted is a Personal View and this Stored Procedure parameter is set to 1 and the User Identifier specified by the @UserId Stored Procedure parameter originally created the View, then the View MUST be deleted. If the specified View to be deleted is a Personal View and this Stored Procedure parameter is set to 0, then the View MUST NOT be deleted.

@CanManageLists: If the specified View to be deleted is a shared view and this Stored Procedure parameter is set to 1, then the View MUST be deleted. If the specified View to be deleted is a Shared View and this Stored Procedure parameter is set to 0, then the View MUST NOT be deleted.

@UserId: The User Identifier that originally created the specified View. If the specified View to be deleted is a Personal View and the User Identifier specified by this Stored Procedure parameter is the original creator of the View and the @CanManagePersonalViews Stored Procedure parameter is set to 1, then the View MUST be deleted. If the specified View to be deleted is a Personal View and
the User Identifier specified by this Stored Procedure parameter is not the original creator of the View, then the View MUST NOT be deleted.

Return Code Values: An integer which MUST be listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful execution.</td>
</tr>
<tr>
<td>3</td>
<td>The View specified by the @ViewId Stored Procedure parameter does not exist or the View specified by the @ViewId Stored Procedure parameter is not the current version.</td>
</tr>
<tr>
<td>5</td>
<td>This Return Code MUST be returned if any of the follow occur:</td>
</tr>
<tr>
<td></td>
<td>• The View specified by the @ViewId Stored Procedure parameter exists but is not a View for the List specified by the @ListId Stored Procedure parameter.</td>
</tr>
<tr>
<td></td>
<td>• The View specified by the @ViewId Stored Procedure parameter is a Personal View, and the @CanManagePersonalViews Stored Procedure parameter is not 1.</td>
</tr>
<tr>
<td></td>
<td>• The View specified by the @ViewId Stored Procedure parameter is a Personal View, and the User Identifier specified by the @UserId Stored Procedure parameter is not the original creator of the specified View to be deleted.</td>
</tr>
<tr>
<td></td>
<td>• The View specified by the @ViewId Stored Procedure parameter is a Shared View, and the @CanManageLists Stored Procedure parameter is not 1.</td>
</tr>
</tbody>
</table>

Result Sets: MUST NOT return any Result Sets.

3.1.4.30 proc_DeleteWeb

The proc_DeleteWeb Stored Procedure is called to delete a Site. The T-SQL syntax for the Stored Procedure is as follows:

```
PROCEDURE proc_DeleteWeb(
    @WebSiteId uniqueidentifier,
    @WebUrl nvarchar(260),
    @FailedUrl nvarchar(260) = NULL OUTPUT,
    @DeleteFlags int = 0,
    @WebIdDelete uniqueidentifier = NULL
);
```

@WebSiteId: The Site Collection Identifier of the Site Collection that contains the specified Site to be deleted.

@WebUrl: The URL of the Site in Store-Relative Form to be deleted.

@FailedUrl: If this Stored Procedure parameter is not NULL and there is a failure when deleting the Site, this parameter MAY be assigned the URL in Store-Relative Form of the Document that failed to be deleted from the Site.

@DeleteFlags: The delete flags used to perform additional operations for the Site to be deleted. MUST be one of the Delete Flags defined in Delete Flags (Section 2.2.2.4). If this Stored Procedure parameter is 0, then it is ignored and the Site is simply deleted.
@WebIdDelete: If this Stored Procedure parameter is not NULL, then this is the Site Identifier of the Site that will be deleted if a Site does not exist at the URL specified by the @WebUrl Stored Procedure parameter.

Return Code Values: An integer which MUST be listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful execution.</td>
</tr>
<tr>
<td>3</td>
<td>This return code is returned if a site does not exist at the URL specified by the @WebUrl stored procedure parameter or the site collection specified by the @WebSiteId stored procedure parameter does not exist.</td>
</tr>
<tr>
<td>5</td>
<td>This return code is returned if the site exists at the URL specified by the @WebUrl stored procedure parameter, but it does not have a parent site and the @DeleteFlags stored procedure parameter did not specify 8 as its delete flags.</td>
</tr>
</tbody>
</table>
| 33    | The site deletion operation MUST fail with this return code if all of the following are true:  
  - The URL specified by the @WebUrl stored procedure parameter is an existing folder.  
  - The @DeleteFlags stored procedure parameter did not specify 8 as its delete flags.  
  - The existing folder or any of its child folders contain documents that are checked out. |
| 161   | The site deletion operation MUST fail with this return code if the following is true:  
  - The URL specified by the @WebUrl stored procedure parameter is an existing folder.  
  - The @DeleteFlags stored procedure parameter did not specify 8 as its delete flags. |
| 138   | The site deletion operation MUST fail with this return code if all of the following are true:  
  - The URL specified by the @WebUrl stored procedure parameter is an existing folder.  
  - The @DeleteFlags stored procedure parameter did not specify 8 as its delete flags.  
  - The existing folder or any of its child folders is already contained within a list. |
| 206   | The site deletion operation MUST fail with this return code if the following is true:  
  - The URL specified by the @WebUrl stored procedure parameter is an existing folder.  
  - The @DeleteFlags stored procedure parameter did not specify 8 as its delete flags.  
  - The existing folder or any of its child folders exceeds the maximum directory name of 256 Unicode characters. |

Result Sets: MUST return the Audit Flags Result Set.

3.1.4.30.1 Audit Flags Result Set

The proc_DeleteWeb Stored Procedure returns the Audit Flags for the deleted Site and for the Site Collection that contains the Site. The Audit Flags Result Set MUST return one row. The T-SQL syntax for the result set is as follows:
{WebId}: The Site Identifier of the Site that was deleted.
{WebAuditFlags}: The Audit Flags of the Site that was deleted.
{WebInheritAuditFlags}: The Audit Flags that are inherited from the deleted Site's Parent Site.
{SiteAuditFlags}: The Audit Flags of the Site Collection that contains the deleted Site.

3.1.4.31 proc_DropListRecord

The proc_DropListRecord Stored Procedure is called to delete a List Item and optionally place the List Item in the Recycle Bin. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_DropListRecord(
    @SiteId                  uniqueidentifier,
    @WebId                   uniqueidentifier,
    @ListId                  uniqueidentifier,
    @ServerTemplate          int,
    @Id                      int,
    @UseNvarchar1ItemName    bit = 1,
    @AuditIfNecessary       bit= 0,
    @UserTitle               nvarchar(255),
    @Version                 int= NULL,
    @UserId                  int= 0,
    @NeedsAuthorRestriction  bit= 0,
    @Basetype                int= NULL,
    @DeleteOp                int= 3,
    @eventData               image= NULL,
    @acl                     image= NULL,
    @DeleteTransactionId     varbinary(16)= 0x OUTPUT,
    @Size                    bigint= 0 OUTPUT)
);
```

@SiteId: The Site Collection Identifier of the Site Collection which contains the specified List Item to be deleted.

@WebId: The Site Identifier of the Site containing the List Item to be deleted. This parameter MUST correspond to a valid Site, and MUST NOT be NULL.

@ListId: The List Identifier of the List containing the List Item to be deleted.

@ServerTemplate: The integer value of the List Template of the List that contains the List Item to be deleted.

@Id: The Item identifier of the List Item to be deleted.

@UseNvarchar1ItemName: A bit flag specifying whether to use the nvarchar1 column value of the List Item as the List Item's Display Name.

@AuditIfNecessary: A bit flag specifying whether to audit the delete operation.
@UserTitle: The Display Name of the Current User. This parameter SHOULD be ignored.

@Version: An OPTIONAL value to compare with the internal version number of the List Item. It MAY be NULL. If this parameter is not NULL, the parameter MUST match the internal version number for successful completion.

@UserId: The integer identifier of the Current User. This value MUST NOT be NULL.

@NeedsAuthorRestriction: A bit flag specifying whether only the list item's author is permitted to delete the list item. It MUST NOT be NULL. If this parameter is set to "1", the current user specified by @UserId MUST be the list item's author for successful execution.

@Basetype: The list base type of the list containing the list item. This parameter SHOULD be ignored. The value MUST be listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Generic List</td>
</tr>
<tr>
<td>1</td>
<td>Document Library</td>
</tr>
<tr>
<td>3</td>
<td>Discussion Board</td>
</tr>
<tr>
<td>4</td>
<td>Survey List</td>
</tr>
<tr>
<td>5</td>
<td>Issues List</td>
</tr>
</tbody>
</table>

@DeleteOp: A parameter specifies the delete option. The value MUST be listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>The deleted List Item MUST NOT be placed in the Recycle Bin (non-recoverable delete).</td>
</tr>
<tr>
<td>4</td>
<td>The deleted List Item MUST be placed in the Recycle Bin (recoverable delete).</td>
</tr>
</tbody>
</table>

@eventData: Contains implementation-specific event data significant to the front-end Web server but otherwise opaque to the Back-End Database Server to be stored by the Back-End Database Server. It MAY be NULL.

@acl: The binary serialization of the Access Control List (ACL) Format access control list for the data supplied in @eventData, to be stored with the data. It MAY be NULL.

@DeleteTransactionId: A GUID which identifies the transaction that encapsulated the actual delete operation. This is used so that multiple or hierarchical operations MAY be performed by the caller. If this is a zero length binary (0x) and @DeleteOp is set to 4, this procedure will define this as a new 16-byte Identifier (converted GUID) and add a record to the Recycle Bin, otherwise it will use the passed in value in its work.

@Size: The size of the List Item. It MUST NOT be NULL. The proc_DropListRecord MUST return the number of bytes used by the List Item through the @Size parameter if the execution succeeded.

Return Code Values: An integer which MUST be listed in the following table:
<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful execution.</td>
</tr>
<tr>
<td>3</td>
<td>The list item does not exist.</td>
</tr>
<tr>
<td>5</td>
<td>Access denied. The current user specified by @UserId parameter is not same as the author of the list item when @NeedsAuthorRestriction is &quot;1&quot;.</td>
</tr>
<tr>
<td>33</td>
<td>Attempt to delete directories that contain checked out files.</td>
</tr>
<tr>
<td>1150</td>
<td>Concurrency violation. The @Version parameter does not match the internal version number of the List Item. The proc_DropListRecord MUST only return this value if @Version is not NULL.</td>
</tr>
</tbody>
</table>

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

### 3.1.4.32 proc_FetchOldDoc

The proc_FetchOldDoc Stored Procedure is called to return a Historical Version of a Document for the **HTTP GET** and **HTTP HEAD** operations for a specified User. Different sets of information are provided depending on the type of request (HTTP GET or HTTP HEAD). The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_FetchOldDoc(
    @DocSiteId uniqueidentifier,
    @DocDirName nvarchar(256),
    @DocLeafName nvarchar(128),
    @IfModifiedSince datetime,
    @FetchType int,
    @ValidationType int,
    @ClientVersion int,
    @ClientId uniqueidentifier,
    @SystemID varbinary(512),
    @VirusVendorID int,
    @ChunkSize int,
    @DGCacheVersion bigint
);
```

- **@DocSiteId**: The identifier of the Site Collection containing the Document.
- **@DocDirName**: The Directory Name of the Document.
- **@DocLeafName**: The Leaf Name of the Document.
- **@IfModifiedSince**: This parameter is used in combination with @ValidationType to determine whether the Document stream will be returned. If the front-end Web server has a cached copy of the Document stream, @IfModifiedSince SHOULD be the time in Coordinated Universal Time (UTC) the cached copy of the Document was last modified. Otherwise, @IfModifiedSince SHOULD be NULL.
- **@FetchType**: This parameter specifies the type of HTTP request. If set to 0, this specifies an HTTP GET request. If set to 1, this specifies an HTTP HEAD request. All values other than 1 MUST be treated as 0.
- **@ValidationType**: This parameter is used to determine whether the Document stream will be returned. It MUST be listed in the following table:
<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Return Document stream.</td>
</tr>
<tr>
<td>1</td>
<td>Return Document stream if @ClientId does not match the Document Identifier of the Document in the Back-End Database Server.</td>
</tr>
<tr>
<td>2</td>
<td>Return Document stream if @IfModifiedSince does not match the last modified time of the Document in the Back-End Database Server.</td>
</tr>
<tr>
<td>3</td>
<td>Return Document stream if @IfModifiedSince does not match the last modified time of the Document or @ClientId does not match the Document Identifier of the Document in the Back-End Database Server.</td>
</tr>
</tbody>
</table>

@ClientVersion: The User Interface (UI) Version of the requested Document.

@ClientId: The Document Identifier of the Document used as an HTTP entity tag for client cache validation.

@SystemID: The SystemID of the User originating the request; NULL indicates an Anonymous User.

@VirusVendorID: This parameter specifies the identifier of the virus scanner registered for the farm.

@ChunkSize: Specifies the maximum size requested, in bytes, of the Document. If the Document size is larger than this maximum size, a single 0 byte is returned for {Content} in the Document Version Content Stream Result Set and the front-end Web server MAY request the remainder of the Document in a subsequent operation.

@DGCacheVersion: The version of the domain group cache as seen by the front-end Web server. It is used to compare with the Domain Group cache version in the Back-End Database Server to determine whether an update is needed. A special value of -2 is specified to indicate that the version numbers of the Domain Group cache are not requested.

Return Values: proc_FetchOldDoc returns an integer Return Code which MUST be listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful execution.</td>
</tr>
<tr>
<td>3</td>
<td>The Current Version of the Document was not found. Or The specified Historical Version of the Document could not be found, or the requested Version of the Document is the Current Version of the Document for the specified User.</td>
</tr>
<tr>
<td>18</td>
<td>The client's cached copy of the Document SHOULD be used.</td>
</tr>
</tbody>
</table>

Result Sets: proc_FetchOldDoc MUST return the following Result Sets conditionally as described in each Result Set section.

3.1.4.32.1 Domain Group Cache Versions Result Set

The Domain Group Cache Versions Result Set returns the Version numbers associated with the Domain Group cache for the Site Collection containing the Document. The Domain Group cache
contains a serialized binary representation of the **external groups** that are Members of the **Site Groups**.

The Domain Group Cache Versions Result Set MUST contain one row. If @DGCacheVersion is -2, then all Columns returned will have the Value -2 as well. The Domain Group Cache Versions Result Set is defined in the proc_SecGetDomainGroupMapData.Domain Group Cache Versions Result Set section.

### 3.1.4.32.2 Domain Group Cache Back-End Database Server Update Result Set

The Domain Group Cache Back-End Database Server Update Result Set contains information to be used in re-computing the Domain Group cache.

The Domain Group Cache Back-End Database Server Update Result Set returns only if @DGCacheVersion is not -2 and the real Version of the Domain Group is more recent than the cached Version (The Value of RealVersion is greater than the Value of CachedVersion in the Domain Group Cache Versions Result Set).

If the Domain Group Cache Back-End Database Server Update Result Set is returned, it indicates that the database's copy of the **domain** Group cache is out of date and **SHOULD** be recomputed to ensure that proper security checks can be made. When returned, the Domain Group Cache Back-End Database Server Update Result Set MUST have one row. The Domain Group Cache Back-End Database Server Update Result Set is defined in the proc_SecGetDomainGroupMapData. Domain Group Cache Back-End Database Server Update Result Set section.

### 3.1.4.32.3 Domain Group Cache Front-End Web Server Update Result Set

The Domain Group Cache Front-End Web Server Update Result Set contains the binary data needed to refresh the Domain Group cache in the front-end Web server.

The Domain Group Cache Front-End Web Server Update Result Set returns only if @DGCacheVersion is not -2 and the cached Version is up to date (the Value of RealVersion is not greater than the Value of CachedVersion in the Domain Group Cache Versions Result Set).

The Domain Group Cache Front-End Web Server Update Result Set is defined in the proc_SecGetDomainGroupMapData. Domain Group Cache Update Result Set section.

### 3.1.4.32.4 Document Version Metadata Result Set

The Document Version Metadata Result Set returns the Document Metadata for the specified Version. If the Current Version of the Document is not found, this Result Set MUST NOT be returned. If the specified version of the Document is not found, this Result Set MUST be returned with zero rows. The T-SQL syntax for the result set is as follows:

```sql
Size int,
DocFlags int,
TimeCreated datetime,
FullUrl nvarchar(260),
{WebId} uniqueidentifier,
FirstUniqueAncestorWebId uniqueidentifier,
SecurityProvider uniqueidentifier,
{InDocLibrary} bit,
{DocId} uniqueidentifier,
{SiteFlags} int,
Acl image,
AnonymousPermMask bigint,
```

---

*Windows SharePoint Services: Content Database Document and List Item Management Communications Protocol Specification*

Copyright © 2012 Microsoft Corporation.

Release: July 16, 2012
tp_ID                           uniqueidentifier,
 tp_Id                           int,
 tp_SiteAdmin                    bit,
 tp_IsActive                     bit,
 tp_Login                        nvarchar(255),
 tp_Email                        nvarchar(255),
 tp_Title                        nvarchar(255),
 tp_Notes                        nvarchar(1023),
 tp_ExternalTokenLastUpdated     datetime,
 tp_Token                        image,
 UserId                          int,
 {SiteSecurityVersion}           bigint,
 {PermCheckedAgainstUniqueList}  int,
 DraftOwnerId                    int,
 tp_Flags                        bigint,
 Level                           tinyint,
 {VirusVendorID}                 int,
 {VirusStatus}                   int,
 {VirusInfo}                     nvarchar(255),
 {ContentModifiedSince}          bit,
 {ProgId}                        nvarchar(255),
 {DirName}                       nvarchar(128),
 {LeafName}                      nvarchar(128),
 {Type}                          tinyint;

**Size:** The size in bytes of the specified Version of the Document.

**DocFlags:** The Document Flags value of the document version.

**TimeCreated:** The date and time in Coordinated Universal Time (UTC) when the Document for the specified Version was last modified.

**FullUrl:** The Store-Relative Form URL of the Document.

**{WebId}:** The Site IDENTIFIER of the Site containing the Document Version.

**FirstUniqueAncestorWebId:** The Site IDENTIFIER whose security permissions are effective for the Site containing the specified Document.

**SecurityProvider:** The identifier of the COM class of the security provider (1) for this Site. This MUST be NULL for Sites using the native security implementation.

**{InDocLibrary}**: If the Document is in a Document Library, this MUST be set to 1; otherwise, it MUST be set to 0.

**{DocId}**: The Document Identifier of the requested Document.

**{SiteFlags}**: The Site Collection Flags value describing the configuration of the Site Collection containing the Document.

**Acl:** The binary serialization of the Access Control List (ACL) that is effective for the Document.

**AnonymousPermMask:** Contains a 64-bit mask that specifies the Permissions granted to an Anonymous User.

**tp_ID:** The List IDENTIFIER of the List containing the Document Version. It MUST be NULL if this Document is not in a List.
**tp_Id**: The User Identifier for the specified User.

**tp_SiteAdmin**: Indicates whether the specified User is an Administrator of the Site Collection. If yes, this MUST be set to 1; otherwise, this MUST be set to 0.

**tp_IsActive**: Indicates whether the specified User has created or modified any data in the Site Collection. If yes, this MUST be set to 1; otherwise, this MUST be set to 0.

**tp_Login**: The Login Name of the specified User.

**tp_Email**: The E-mail address of the specified User.

**tp_Title**: The Display Name of the specified User.

**tp_Notes**: Notes about the specified User.

**tpExternalTokenLastUpdated**: The date and time in Coordinated Universal Time (UTC) when the External Group Token value for the specified User was last updated. Refer to [MS-WSSFO], section 2.2.4.2.

**tp_Token**: An External Group Token value encoding information about External Group membership for the User. This value can be NULL, indicating that this User has never visited any Site in the Site Collection. If this value is NULL, the value in tpExternalTokenLastUpdated MUST also be NULL. Refer to [MS-WSSFO], section 2.2.4.2.

**UserId**: The User Identifier of the specified User. This MAY be NULL if the User has not been added as a Member to the Site whose Permissions are in effect on the Document.

**{SiteSecurityVersion}**: A version number incremented when changes are made to the Site Collection's permissions.

**{PermCheckedAgainstUniqueList}**: This MUST be 0.

**DraftOwnerId**: If the Document Version is a Draft, the User Identifier of the User who saved the first Draft after the previous published version MUST be returned. If the Document Version is a Published Version and if there exists a Draft whose version number is between that of the requested Version and the next Published Version, the User Identifier of the User who saved the first Draft after the requested Version MUST be returned. In all other cases, NULL MUST be returned.

**tp_Flags**: A list flags value describing the List that contains the Document.

**Level**: A Publishing Level value specifying the publishing status of this Document Version.

**{VirusVendorID}**: The identifier of the Virus Scanner which processed this Document Version. This value MUST be NULL if the Document has not been processed by a Virus Scanner.

**{VirusStatus}**: An enumerated type specifying the current virus check status of this Document Version. This value MUST be NULL if the requested Document Version has not been processed by a virus scanner. See [MS-WSSFO], section 2.2.3.15 for a list of valid values.

**{VirusInfo}**: A string containing a provider-specific message returned by the Virus Scanner when it last processed the Document Version.

**{ContentModifiedSince}**: A bit indicating if the Document Version has been modified, depending on the value of @ValidationType.

It SHOULD<6> be set to 1 if any of the following is true:
The document is a dynamic page.

- @ValidationType is 1 and the value of @ClientId does not match the Document IDENTIFIER in the store;
- @ValidationType is 2 and the time the specified Document Version was last modified does not match @IfModifiedSince.
- @ValidationType is 3 and the time the specified Document Version was last modified does not match @IfModifiedSince OR the value of @ClientId does not match the Document IDENTIFIER in the store.

In all other cases, it MUST be set to zero.

{ProgId}: Designates a preferred application to open the document.

{DirName}: The directory name of the requested document.

{LeafName}: The leaf name of the requested document.

{Type}: The document store type of this document version. Refer to [MS-WSSFO], section 2.2.4.

### 3.1.4.32.5 Document Version Content Stream Result Set

The Document Version Content Stream Result Set contains the specified Document Version's Document Stream and associated Metadata. It MUST return only if @FetchType is not set to 1 (that is, not indicating an HTTP HEAD-only request). Also, it MUST NOT return if either the Current Version of the Document is not found or the specified Historical Version of the Document for the specified User is not found.

The Document Content Stream Result Set MUST return 0 or 1 Rows.

The Result Stream MUST return the Document Stream of the Document if any of the following conditions are set to true:

- The value of {ContentModifiedSince} in the Document Version Metadata Result Set is 1.
- @VirusVendorID is NOT NULL and does not match the value of the identifier of the Virus Scanner that processed the Document and the virus check status associated with the Document is either unknown or set to 0.

Otherwise, the Return Code of 18 MUST be returned and 0 Rows MUST be returned in this Result Set. The T-SQL syntax for the result set is as follows:

```
{Content}            image,
{Size}               int,
{ClientVersion}      int,
{DocId}              uniqueidentifier;
```

- {Content}: The Document Stream of the Document. For an uncustomized document, this MUST be NULL. Otherwise, if the content size, in bytes, is larger than the value specified by @ChunkSize, a single 0 byte MUST be returned, and the front-end Web server MAY request individual Chunks of the Document Stream in subsequent requests.
- {Size}: The Document size, in bytes.
{DocId}: The Document IDENTIFIER of the Document whose Historical Version is being returned.

3.1.4.33  proc_FindDocs

The proc_FindDocs Stored Procedure is called to determine if one or more Documents exist in a Site Collection. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_FindDocs(
    @SiteId       uniqueidentifier,
    @DirName1     nvarchar(256) = NULL,
    @LeafName1    nvarchar(256) = NULL,
    @DirName2     nvarchar(256) = NULL,
    @LeafName2    nvarchar(256) = NULL,
    @DirName3     nvarchar(256) = NULL,
    @LeafName3    nvarchar(256) = NULL,
    @DirName4     nvarchar(256) = NULL,
    @LeafName4    nvarchar(256) = NULL,
    @DirName5     nvarchar(256) = NULL,
    @LeafName5    nvarchar(256) = NULL,
    @DirName6     nvarchar(256) = NULL,
    @LeafName6    nvarchar(256) = NULL,
    @DirName7     nvarchar(256) = NULL,
    @LeafName7    nvarchar(256) = NULL,
    @DirName8     nvarchar(256) = NULL,
    @LeafName8    nvarchar(256) = NULL);
```

@SiteId: The Site Collection Identifier of the Site Collection which contains the specified Documents to be searched.

@DirName1: The Store-Relative Form folder URL that contains the Document specified by @LeafName1.

@LeafName1: The Document name in the folder specified by @DirName1 that is to be retrieved.

@DirName2: The Store-Relative Form folder URL that contains the Document specified by @LeafName2.

@LeafName2: The Document name in the folder specified by @DirName2 that is to be retrieved.

@DirName3: The Store-Relative Form folder URL that contains the Document specified by @LeafName3.

@LeafName3: The Document name in the folder specified by @DirName3 that is to be retrieved.

@DirName4: The Store-Relative Form folder URL that contains the Document specified by @LeafName4.

@LeafName4: The Document name in the folder specified by @DirName4 that is to be retrieved.

@DirName5: The Store-Relative Form folder URL that contains the Document specified by @LeafName5.

@LeafName5: The Document name in the folder specified by @DirName5 that is to be retrieved.

@DirName6: The Store-Relative Form folder URL that contains the Document specified by @LeafName6.
@LeafName6: The Document name in the folder specified by @DirName6 that is to be retrieved.

@DirName7: The Store-Relative Form folder URL that contains the Document specified by @LeafName7.

@LeafName7: The Document name in the folder specified by @DirName7 that is to be retrieved.

@DirName8: The Store-Relative Form folder URL that contains the Document specified by @LeafName8.

@LeafName8: The Document name in the folder specified by @DirName8 that is to be retrieved.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST return 1 result set as follows:

3.1.4.33.1 Found Docs Result Set

Found Docs Result Set returns the full URL of each Document specified in the request that was found in the Site Collection. The Found Docs Result Set returns one row for each found Document. The T-SQL syntax for the result set is as follows:

```
FullName nvarchar(384);
```

FullName: The Store-Relative Form URL for the specified Document.

3.1.4.34 proc_FinishUndirtyList

The proc_FinishUndirtyList Stored Procedure is called to reset the List Flag to "not Dirty " on a List upon completion of a bulk row update operation. The T-SQL syntax for the Stored Procedure is as follows:

```
PROCEDURE proc_FinishUndirtyList(
    @WebId uniqueidentifier,
    @ListId uniqueidentifier,
    @CacheParseId uniqueidentifier,
    @ListFlag int
);
```

@WebId: The Site Identifier for the Site Collection containing the List. SHOULD NOT be NULL. If @WebId is NULL, the procedure MUST return with no error and MUST have no effect.

@ListId: The list Identifier of the list to reset the "dirty" status. SHOULD NOT be NULL. If it is NULL, the procedure will not return an error, but it will have no effect.

@CacheParseId: Used for concurrency detection when two different requests attempt to reset the dirtying on a List or its Documents at the same time. Compared to the CacheParseId column in the Lists View. If the List row’s CacheParseId value has changed since the bulk row update operation started, the Stored Procedure will have no effect. The latest bulk row operation will have to reset the dirty status. If @CacheParseId is NULL, the Stored Procedure will not return an error, but it will have no effect.

@ListFlag: Bit containing control flags. The only flag of interest is LDD_NEEDSLINKFIXUP (value = 1). SHOULD NOT be NULL. If it is NULL, the stored procedure will not return an error, but it will have no effect.
Return Code Values: An integer which the protocol client MUST ignore.

Result Sets: MUST NOT return any Result Sets.

3.1.4.35 proc_GenerateUniqueFileName

The proc_GenerateUniqueFileName Stored Procedure is called to generate a unique file name from given base name and extension. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_GenerateUniqueFileName(
    @SiteId           uniqueidentifier,
    @BaseUrl          nvarchar(260),
    @Extension        nvarchar(10),
    @MaxAttempts      int
);
```

@SiteId: The Site Collection Identifier of the Site Collection which is to contain the specified File.

@BaseUrl: The desired base name in Store-Relative Form of the file.

@Extension: The desired extension of the file.

@MaxAttempts: The maximum number of attempts to generate a unique file name.

Return Code Values: An integer which MUST be listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful execution.</td>
</tr>
<tr>
<td>3</td>
<td>No unique file name can be generated within the specified @MaxAttempts.</td>
</tr>
</tbody>
</table>

Result Sets: MUST return 1 result set if the return code is 0 and MUST NOT return a result set if the return code is not 0.

3.1.4.35.1 Unique File Name Result Set

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

```sql
{File Name}    nvarchar(385)
```

{File Name}: The URL in Store-Relative Form, generated from the given base file name and extension, which is unique in the Site Collection.

3.1.4.36 proc_GetAllAttachmentsInfo

The proc_GetAllAttachmentsInfo Stored Procedure is called to retrieve the information of attachments for a List or a List Item. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_GetAllAttachmentsInfo(
    @SiteID      uniqueidentifier,
    @WebID       uniqueidentifier,
    @ListID      uniqueidentifier,
    @ItemID      int
);
@SiteID: The Site Collection Identifier of the Site Collection which contains the specified List or List Item.

@WebID: The Site Identifier of the Site that contains the List or List Item. This parameter MUST correspond to a valid Site, and MUST NOT be NULL.

@ListID: The List Identifier of the List.

@ItemID: The List Item Identifier of the List Item. A value of -1 means to retrieve the information of all attachments of the List and order by List Item Identifier; otherwise retrieve the information of attachments for the specified List Item.

Return Code Values: An integer which MUST be 0.

Return Sets: MUST NOT return any Result Set when the given List does not exist. Otherwise MUST return 1 List Attachments Result Set when the value of @ItemID is -1 or 1 List Item Attachments Result Set when the value of @ItemID is not -1.

### 3.1.4.36.1 List Attachments Result Set

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

```
{ItemID}           int,
LeafName           nvarchar(128),
Id                 uniqueidentifier,
Version            int,
Acl                image,
AnonymousPermMask  bigint;
```

{ItemID}: The Item Identifier of the List Item that has the Attachment.

LeafName: The leaf name of the Attachment.

Id: The identifier of the Attachment.

Version: The Version of the Attachment.

Acl: The Access Control List (ACL) of the Attachment.

AnonymousPermMask: The Permissions of anonymous users for the Attachment. This MUST not be NULL.

### 3.1.4.36.2 List Item Attachments Result Set

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

```
Id            uniqueidentifier,
LeafName      nvarchar(128);
```

Id: The identifier of the Attachment.

LeafName: The leaf name of the Attachment.
3.1.4.37 proc_GetChanges

The proc_GetChanges Stored Procedure is called to get the Events from the Change Log specified by the parameters. The T-SQL syntax for the Stored Procedure is as follows:

```t-sql
PROCEDURE proc_GetChanges(
    @SiteId uniqueidentifier,
    @WebId uniqueidentifier,
    @ListId uniqueidentifier,
    @ChangeTime datetime,
    @ChangeNumber bigint,
    @ChangeTimeEnd datetime,
    @ChangeNumberEnd bigint,
    @ObjectTypeMask int,
    @EventTypeMask int
);
```

@SiteId: The Site Collection Identifier of the Site Collection with which the Events are associated. If this parameter is NULL, Events from all Site Collections MUST be included.

@WebId: The Site Identifier of the Site.

@ListId: The List Identifier of the List with which the Events are associated. If this parameter is NULL, then all Events in the Change Log that have an empty Site Identifier MUST be included.

@ChangeTime: A timestamp in Coordinated Universal Time (UTC). This parameter defines the lower bound timestamp of the Events returned from the Change Log. If this parameter is NULL, then Events with change log identifier greater than or equal to 0 MUST be included.

@ChangeNumber: The lower bound Change Log Identifier of the Events to be included in the result. If this parameter is non-NULL, then the @ChangeTime parameter MUST be ignored.

@ChangeTimeEnd: A Time Stamp in Coordinated Universal Time (UTC). This parameter defines the upper bound timestamp of the Events returned from the Change Log. If this parameter is NULL, then the upper bound will be the most recent Event in the Change Log.

@ChangeNumberEnd: The upper bound Change Log Identifier of the Events returned from the Change Log. If this parameter is non-NULL, then the @ChangeTimeEnd parameter MUST be ignored.

@ObjectTypeMask: A 4-byte unsigned integer bit mask that specifies the type of objects upon which an Event had happened. Valid values of this flag are defined in Event Object Type Flags (Section 2.2.2.1). One or more flags could be set in this parameter.

@EventTypeMask: A 4-byte integer bit mask that specifies the type of an Event. This parameter MAY have one or more flags set. Valid values of this flag are defined in Bit Fields and Flag Structures (Section 2.2.2.2).

Return Code Values: An integer which the protocol client MUST ignore.

Result Sets: MUST return two Result Sets in the following order:

3.1.4.37.1 EventInformation Result Set

The EventInformation Result Set returns the Event that has the smallest Change Log Identifier in the Change Log. The EventInformation Result Set MUST return one row in the Result Set if an Event
exists in the Change Log or zero rows if no Event exists in the Change Log. The T-SQL syntax for the result set is as follows:

```
EventTime    datetime,
Id           bigint;
```

**EventTime:** A timestamp in Coordinated Universal Time (UTC) that specifies the time when this Event occurred.

**Id:** The Change Log Identifier of this Event.

### 3.1.4.37.2 EventDetails Result Set

EventDetails returns details of Events that satisfy the input parameters. The EventDetails Result Set MAY return less rows than the total number of Events found. The T-SQL syntax for the result set is as follows:

```
EventTime            datetime,
Id                   bigint,
SiteId               uniqueidentifier,
WebId                uniqueidentifier,
ListId               uniqueidentifier,
ItemId               int,
DocId                uniqueidentifier,
Guid0                uniqueidentifier,
Int0                 int,
ContentTypeId        varbinary(512),
ItemFullUrl          nvarchar(266),
EventType            int,
ObjectType           int,
TimeLastModified     datetime,
Int1                 int;
```

**EventTime:** A timestamp in Coordinated Universal Time (UTC) that specifies when this Event occurred. Valid values are defined in Event Object Type Flags (section 2.2.2.1).

**Id:** The Change Log Identifier of this Event. Valid values are defined in Event Object Type Flags (section 2.2.2.1).

**SiteId:** This value is a Change Log SiteId (section 2.2.1.11).

**WebId:** This value is a Change Log WebId (section 2.2.1.12).

**ListId:** This value is a Change Log ListId (section 2.2.1.1).

**ItemId:** This value is a Change Log ItemId (section 2.2.1.2).

**DocId:** This value is a Change Log DocId (section 2.2.1.2).

**Guid0:** This value is a Change Log Guid0 (section 2.2.1.4).

**Int0:** This value is a Change Log Int0 (section 2.2.1.4).

**ContentTypeId:** This value is a Change Log ContentTypeId (section 2.2.1.6).

**ItemFullUrl:** This value is a Change Log ItemFullUrl (section 2.2.1.7)
**EventType:** A 4-byte unsigned integer bit mask that specifies the type of an Event. Valid values of this flag are defined in Bit Fields and Flag Structures (section 2.2.2).

**ObjectType:** A 4-byte integer bit mask that specifies the type of object upon which an Event had happened. Valid values of this flag are defined in Simple Data Types (section 2.2.1).

**TimeLastModified:** This value is specified in Change Log TimeLastModified (section 2.2.1.8)

**Int1:** This value is a Change Log Int1 (section 2.2.1.10)

### 3.1.4.38 proc_GetCurrent

The proc_GetCurrent Stored Procedure is called to return the timestamp and Change Log Identifier of the latest Event from the Change Log. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_GetCurrent();
```

**Return Code Values:** An integer which the protocol client MUST ignore.

**Result Sets:** MUST return one Result Set as follows:

#### 3.1.4.38.1 EventInformation Result Set

EventInformation returns the timestamp and Change Log Identifier of the latest Event from the Change Log. The EventInformation Result Set MUST return one row if an Event is found or zero rows if no Event is found. The T-SQL syntax for the result set is as follows:

```sql
EventTime    datetime,
Id           bigint;
```

**EventTime:** A timestamp in Coordinated Universal Time (UTC) that specifies when this Event occurred.

**Id:** The Change Log Identifier of this Event.

### 3.1.4.39 proc_GetDocIdUrl

The proc_GetDocIdUrl Stored Procedure is called to retrieve the IDENTIFIER of a specified Document. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_GetDocIdUrl(
    @SiteID           uniqueidentifier,
    @DocDirName       nvarchar(256),
    @DocLeafName      nvarchar(128),
    @DocID            uniqueidentifier OUTPUT
);
```

**@SiteId:** The Site Collection Identifier of the Site Collection which contains the specified Document.

**@DocDirName:** The directory name of the specified Document.

**@DocLeafName:** The leaf name of the requested Document.
@DocID: An output parameter containing the IDENTIFIER of the specified Document if execution is successful. If the specified Document does not exist, the proc_GetDocIdUrl MUST NOT set the value of @DocID.

Return Code Values: An integer which MUST be listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful execution.</td>
</tr>
<tr>
<td>2</td>
<td>Cannot find the specified Document in the Site Collection, or the Site Collection does not exist.</td>
</tr>
</tbody>
</table>

Result Sets: MUST NOT return any Result Sets.

3.1.4.40 proc_GetFullLinkInfoForSingleDoc

The proc_GetFullLinkInfoForSingleDoc Stored Procedure is called to return information of all the links for a single Document. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_GetFullLinkInfoForSingleDoc(
    @DocSiteId                   uniqueidentifier,
    @DocDirName                  nvarchar(256),
    @DocLeafName                 nvarchar(128),
    @UserId                      int,
    @AttachmentsFlag             tinyint,
    @MaxCheckinLevel             tinyint,
    @GetWebListForNormalization  bit );
```

@DocSiteId: The Site Collection Identifier of the Site Collection containing the Document. MUST NOT be NULL.

@DocDirName: The directory name of the directory containing the Document. MUST NOT be NULL.

@DocLeafName: The leaf name of the Document. MUST NOT be NULL.

@UserId: Identifier of the current user. MUST NOT be NULL.

@AttachmentsFlag: Bit that governs the type of security checks which SHOULD be performed by a Stored Procedure on this Document's URL, based on whether it appears to be an Attachment. A value which MUST be listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>The URL does not appear to be an Attachment.</td>
</tr>
<tr>
<td>1</td>
<td>The URL is an Attachment file. The Directory Name of the Document has the string &quot;Attachments&quot; as its next-to-last path segment, and a 32-bit base-10 signed integer as the last path segment that is referring to the item identifier to which this file is attached and where the permissions will be checked. For example, &quot;Announcements/Attachments/17/file1.txt&quot;.</td>
</tr>
<tr>
<td>2</td>
<td>The URL is a List Item Attachment Folder. For example, &quot;Announcements/Attachments/17&quot;.</td>
</tr>
<tr>
<td>3</td>
<td>The URL is the List Attachment Folder itself. The last path segment of the URL is the string &quot;Attachments&quot;. For example, &quot;Announcements/Attachments&quot;.</td>
</tr>
</tbody>
</table>
@MaxCheckinLevel: The maximum Publishing Level of the links to return. MUST be one of these values:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>&quot;UNUSED&quot;</td>
</tr>
<tr>
<td>1</td>
<td>&quot;DEFAULT&quot; or &quot;PUBLISH&quot;</td>
</tr>
<tr>
<td>2</td>
<td>&quot;Draft&quot;</td>
</tr>
<tr>
<td>255</td>
<td>&quot;CHECKOUT&quot;</td>
</tr>
</tbody>
</table>

@GetWebListForNormalization: Bit flag indicating that the Web List For Normalization Result Set SHOULD be returned.

Return Code Values: An integer which MUST be listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Success</td>
</tr>
<tr>
<td>2</td>
<td>The Document store type of the specified object is not 0, indicating that it is not a Document.</td>
</tr>
</tbody>
</table>

Result Sets: The Stored Procedure MAY return the Web List For Normalization Result Set; and MUST return the Individual URL Security Result Set, the Document Link Information Result Set, and the Document Setup Path Result Set.

3.1.4.40.1 Web List For Normalization Result Set

If the @GetWebListForNormalization flag is set to 'True', this result set returns a list of Full URL's for subsites of the specified Site. The result set will contain zero or more rows. The T-SQL syntax for the result set is as follows:

```
FullUrl      nvarchar(256);
```

**FullUrl**: Full URL to a subsite.

3.1.4.40.2 NULL Individual URL Security Result Set

The NULL Individual URL Security Result Set MUST ONLY be returned if the specified Document location is NOT contained within a List or Document Library. It MUST contain a single row. The NULL Individual URL Security Result Set is defined in [MS-WSSFO], section 2.2.5.14.

3.1.4.40.3 Individual URL Security Result Set

This Result Set contains security information about the specified Document. If the Document does not exist, but the specified URL is within a List or Document Library, security information is returned from the effective Security Scope for the specified Document location.

The Individual URL Security Result Set MUST ONLY be returned if the specified Document location is contained within a List or Document Library. Otherwise, the NULL Individual URL Security Result Set MUST be returned instead. If returned, the Individual URL Security Result Set MUST contain a single row. The Individual URL Security Result Set is defined in [MS-WSSFO], section 2.2.5.10.
3.1.4.40.4  Document Link Information Result Set

Returns information about each Forward Link from the Document and Backward Link to the Document within the Site Collection. The Result Set MUST be returned and MUST contain one row for each Forward Link within the specified Document, and one row for each Backward Link to the Document within the specified Site Collection. The T-SQL syntax for the result set is as follows:

```
LinkDirName       nvarchar(256),
LinkLeafName      nvarchar(128),
LinkType          tinyint,
LinkSecurity      tinyint,
LinkDynamic       tinyint,
LinkServerRel     bit,
LinkStatus        tinyint,
PointsToDir       bit,
WebPartId         int,
LinkNumber        int,
WebId             uniqueidentifier,
Search            ntext,
FieldId           uniqueidentifier;
```

**LinkDirName:** The directory name of the directory containing the linked object. This value MUST NOT be NULL.

**LinkLeafName:** The leaf name of the linked object. This value MUST NOT be NULL.

**LinkType:** Type of the Link. Refer to [MS-WSSFO], section 2.2.3.8, for valid values.

**LinkSecurity:** A one-byte (tinyint) value represented as a single upper case ASCII character specifying the Link's URL type. Refer to [MS-WSSFO], section 2.2.3.8 for valid values.

**LinkDynamic:** A one-byte (tinyint) value represented as a single upper case ASCII character which tracks various special Link Types. MUST be one of these values:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>68 (D)</td>
<td>The URL is &quot;dynamic&quot;, which is a Link to &lt;Site URL&gt;/_vti_bin/shtml.dll/DirName/LeafName. Such Links are used to call the FrontPage Smarthtml interpreter on a file.</td>
</tr>
<tr>
<td>71 (G)</td>
<td>A non-absolute Link from an uncustomized document that does not fall into any other category.</td>
</tr>
<tr>
<td>72 (H)</td>
<td>The URL is a history Link; that is, it contains a path segment with the string &quot;,_vti_history&quot;.</td>
</tr>
<tr>
<td>76 (L)</td>
<td>The URL is to a layouts Page; that is, it contains a path segment with the string &quot;,_layouts&quot;.</td>
</tr>
<tr>
<td>83 (S)</td>
<td>The URL is &quot;static&quot;, which is the default, and requires no special handling.</td>
</tr>
</tbody>
</table>

**LinkServerRel:** A bit flag which specifies whether the Link URL is server-relative or not. A value of 1 specifies a server-relative URL. This value MUST be NULL for a Backward Link.

**LinkStatus:** The Document Store Type value of the Document targeted by a Link. This value MUST be 128 for a Backward Link. If the Forward Link target is a Document that does not exist, or if the forward link refers to a target that exists outside the specified Site Collection, or if it refers to a location that could not be verified, this value MUST be NULL.
**PointsToDir:** A bit flag specifying whether the target of the Forward Link was a directory and has been modified to target a **Welcome page**. This value MUST be NULL for a Backward Link. For a Forward Link, if the target is a directory where a Welcome Page is specified, the Link MUST be changed to the URL of the Welcome Page and PointsToDir MUST be set to 1 so that the Link can be distinguished from an explicit Link to the Welcome Page; otherwise this value MUST be 0.

**WebPartId:** MUST be NULL

**LinkNumber:** MUST be NULL

**WebId:** Site Identifier for backward links.

**Search:** Search parameters for backward links. For Forward Links, this is the search portion of the Link: The Link source starting at either the query string signifier '?' or the bookmark signifier '#'.

**FieldId:** If the Link is for a List Item Field within this Document, this is the Field Identifier of the Field to which this the Link belongs.

### 3.1.4.40.5 Document Setup Path Result Set

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

```sql
{DocSetupPath} nvarchar(255);
```

**{DocSetupPath}**: For a document that is now or once was uncustomized, this contains the setup path fragment relative to the base setup path where the content stream of this document can be found. This value MUST be NULL if the document was never uncustomized.

### 3.1.4.41 proc_GetListDataLinks

The proc_GetListDataLinks Stored Procedure is called to get the list of dirty field links for a range of list items in a Site, sorted alphabetically from (@FirstDirName, @FirstLeafName, @FirstLevel) to (@LastDirName, @LastLeafName, @LastLevel), inclusive of the endpoints. The purpose of this Stored Procedure is to finish cleaning up links after a List’s location or other metadata has been updated. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_GetListDataLinks(
    @SiteId uniqueidentifier,
    @WebId uniqueidentifier,
    @FirstDirName nvarchar(256),
    @FirstLeafName nvarchar(128),
    @FirstLevel tinyint,
    @LastDirName nvarchar(256),
    @LastLeafName nvarchar(128),
    @LastLevel tinyint,
    @GetWebListForNormalization bit
);
```

**@SiteId:** The Site Collection Identifier of the Site Collection which contains the specified List.

**@WebId:** The Site Identifier of the Site.

**@FirstDirName:** First allowable directory name to be returned, in alphabetic order.
@FirstLeafName: First allowable leaf name to be returned, in alphabetic order. This does not have to be a member of the directory @FirstDirName.

@FirstLevel: First allowable Publishing Level to be returned, in numeric order. The level does not have to be the level of a leaf in the directory @FirstDirName.

@LastDirName: Last allowable directory name to be returned, in alphabetic order.

@LastLeafName: Last allowable Leaf name to be returned, in alphabetic order. This does not have to be a member of the directory @LastDirName.

@LastLevel: Last allowable Publishing Level to be returned, in numeric order. This level does not have to be the level of a leaf in the directory @LastDirName.

@GetWebListForNormalization: Bit flag indicating that Result Set SHOULD be returned.

Return Code Values: An integer which the protocol client MUST ignore.

Result Sets: MAY return the WebListForNormalization Result Set, and MUST return the List Data Link Information Result Set.

3.1.4.41.1  Web List For Normalization Result Set

If the @GetWebListForNormalization flag is set to 'True', returns a list of URL's for Subsites of the specified Site Collection. This will be an empty result set if there are no Subsites in the Site Collection. The T-SQL syntax for the result set is as follows:

```
FullUrl      nvarchar(256);
```

**FullUrl**: Full URL of a Subsite in the Site Collection.

3.1.4.41.2  List Data Link Information Result Set

Returns the list of dirty field links for a range of list items in a Site, sorted alphabetically from (@FirstDirName, @FirstLeafName, @FirstLevel) to (@LastDirName, @LastLeafName, @LastLevel), inclusive of the endpoints. This Result Set returns only rows where FieldId is not null, and will contain zero or more rows. The Result Set is defined using T-SQL syntax as follows:

```
DirName            nvarchar(256),
LeafName           nvarchar(128),
Level              tinyint,
FieldId            uniqueidentifier,
TargetDirName      nvarchar(256),
TargetLeafName     nvarchar(128),
Type               tinyint,
Security           tinyint,
Dynamic            tinyint,
ServerRel          bit,
Type               tinyint,
PointsToDir        bit;
```

**DirName**: The Directory name of the directory containing the source object of the Link. This value MUST NOT be NULL.

**LeafName**: The leaf name or the source object.
**Level**: The Publishing Level of the source object. MUST be one of these values:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>&quot;UNUSED&quot;</td>
</tr>
<tr>
<td>1</td>
<td>&quot;DEFAULT&quot; or &quot; PUBLISH &quot;</td>
</tr>
<tr>
<td>2</td>
<td>&quot;Draft &quot;</td>
</tr>
<tr>
<td>255</td>
<td>&quot;CHECKOUT&quot;</td>
</tr>
</tbody>
</table>

**FieldId**: Identifier of the field of the source object. This value MUST NOT be NULL.

**TargetDirName**: The directory name of the directory containing the target object of the Link. This value MUST NOT be NULL.

**TargetLeafName**: The leaf name of the linked object. This value MUST NOT be NULL.

**Type**: Type of the Link. MUST be one of these values:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>The Link is from the ACTION attribute of an HTML form tag.</td>
</tr>
<tr>
<td>B</td>
<td>The Link is from the attribute markup of a Bot.</td>
</tr>
<tr>
<td>C</td>
<td>The Link is from an auto-generated table of contents. Agents MAY ignore the Link type when determining unreferenced files within a Site.</td>
</tr>
<tr>
<td>D</td>
<td>The Link references programmatic content, as in the HTML OBJECT or APPLET tags.</td>
</tr>
<tr>
<td>E</td>
<td>The Link is from a cascading style sheet (CSS).</td>
</tr>
<tr>
<td>F</td>
<td>The Link is from the SRC attribute of an HTML FRAME tag.</td>
</tr>
<tr>
<td>G</td>
<td>The Link is to a Dynamic Web Template for the containing Document.</td>
</tr>
<tr>
<td>H</td>
<td>The Link is from an HTML HREF attribute. This MAY also be used as a default Link Type value if a more precise type does not apply.</td>
</tr>
<tr>
<td>I</td>
<td>The Link is to a Document that the containing Document includes via an include Bot.</td>
</tr>
<tr>
<td>J</td>
<td>The Link is from a Field of this List Item.</td>
</tr>
<tr>
<td>K</td>
<td>Identical to 'H', except that the Link contains an HTML bookmark specifier.</td>
</tr>
<tr>
<td>L</td>
<td>The Link is a target in an HTML image map generated from an image map Bot.</td>
</tr>
<tr>
<td>M</td>
<td>The Link is to an image used in an HTML image map generated from an image map Bot</td>
</tr>
<tr>
<td>O</td>
<td>The Link is part of a cross-page URL connection.</td>
</tr>
<tr>
<td>P</td>
<td>The Link is part of the markup of a URL within the source of the containing Document.</td>
</tr>
<tr>
<td>Q</td>
<td>The Link references a cascading style sheet (CSS) Document which provides style information for the containing Document.</td>
</tr>
<tr>
<td>R</td>
<td>The Link is from the Master Page File attribute of the @Page directive in the containing Document.</td>
</tr>
<tr>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>S</td>
<td>The Link is from an HTML SRC attribute.</td>
</tr>
<tr>
<td>T</td>
<td>The Link is to the index file used by a text search Bot on this Page.</td>
</tr>
<tr>
<td>V</td>
<td>The Link is based on the properties of the Document, rather than anything in the Document stream. The Link type is used in tracking the Link between a Site and the master page URL used for the Site.</td>
</tr>
<tr>
<td>X</td>
<td>The Link is from an XML island within an HTML Document.</td>
</tr>
<tr>
<td>Y</td>
<td>The Link references an HTML Document whose HTML BODY tag attributes are used as a template for the attributes of the containing Document’s BODY tag.</td>
</tr>
<tr>
<td>Z</td>
<td>The Link is part of the markup of a URL which exists in a URL Zone in the containing Document, and is consequently not stored within the source of the containing Document.</td>
</tr>
</tbody>
</table>

**Security:** Type of security for the Link. The value MUST be one of the following:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>H</td>
<td>The Link is to an &quot;HTTP:&quot; URL.</td>
</tr>
<tr>
<td>S</td>
<td>The Link is to an &quot;HTTPS:&quot; URL.</td>
</tr>
<tr>
<td>T</td>
<td>The Link is to an &quot;SHTTP:&quot; URL.</td>
</tr>
<tr>
<td>U</td>
<td>The Link transport security is unknown.</td>
</tr>
</tbody>
</table>

[@Dynamic]: A one byte (tinyint) value represented as a single upper case ASCII character which specifies the special Link Types. The value MUST be one of the following:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>The URL is static, which is the default, and requires no special handling.</td>
</tr>
<tr>
<td>D</td>
<td>The URL is dynamic, which is a Link to &lt;Site URL&gt;/_vti_bin/shtml.dll/DirName/LeafName. Such Links are used to start the FrontPage SmarthHTML interpreter on a file.</td>
</tr>
<tr>
<td>L</td>
<td>The URL is to a layouts Page, in other words it contains a path segment with the string &quot;_layouts&quot;.</td>
</tr>
<tr>
<td>H</td>
<td>The URL is a history Link, in other words it contains a path segment with the string &quot;_vti_history&quot;.</td>
</tr>
<tr>
<td>G</td>
<td>A non-absolute Link from an uncustomized document that does not fall into any other category.</td>
</tr>
</tbody>
</table>

**ServerRel:** If 'True' indicates that the @TargetLeafName is file relative (is a directory). This value MUST NOT be NULL.

**Type:** Type of the linked object. This value MUST be one of these values:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>File</td>
</tr>
<tr>
<td>1</td>
<td>Directory</td>
</tr>
</tbody>
</table>
### Value | Description
---|---
2 | Web
128 | Backward Link

**PointsToDir**: 'True' if the Link is to a directory. This value MUST NOT be NULL.

#### 3.1.4.42 proc_GetUrlDocId

The proc_GetUrlDocId Stored Procedure is called to retrieve the directory name and leaf name of a specified Document. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_GetUrlDocId(
    @SiteId       uniqueidentifier,
    @WebId        uniqueidentifier,
    @DocId        uniqueidentifier
);
```

**@SiteId**: The Site Collection Identifier of the Site Collection which contains the specified Document.

**@WebId**: The Site Identifier of the Site which contains the specified Document.

**@DocId**: The Document Identifier of the Document.

**Return Code Values**: An integer return code which MUST be included the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful execution.</td>
</tr>
<tr>
<td>2</td>
<td>The Document does not exist.</td>
</tr>
</tbody>
</table>

**Result Sets**: proc_GetUrlDocId MUST return 1 Directory And Leaf Names **Result Set**.

#### 3.1.4.42.1 Directory And Leaf Names Result Set

The Directory And Leaf Names Result Set contains one row of the directory name and leaf name of the Document whose Document Identifier is @DocId that exists in the Site whose Site Identifier is @WebId that exists in the Site Collection whose Site Collection Identifier is @SiteId. The Directory And Leaf Names Result Set MUST be returned and MUST contain one row if the Document exists. The T-SQL syntax for the result set is as follows:

```sql
DirName       nvarchar(256),
LeafName      nvarchar(128);
```

**DirName**: The store-relative form directory name of the document.

**LeafName**: The leaf name of the document.

#### 3.1.4.43 proc_InsertEventSubscriptionJunctionEntries

The proc_InsertEventSubscriptionJunctionEntries Stored Procedure is called to add junction entries to the junction table for up to 256 entries for Alerts that are either set to be sent immediate or
scheduled (daily or weekly). Immediate alerts send e-mail immediately after an event has taken place. For scheduled subscriptions, the events are generated and stored in a table called junction table and each row called junction entry. When the alert is due, then the e-mail message is generated by combining all the events (junction entries) that matched that particular subscription in the form of a digest and is sent. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_InsertEventSubscriptionJunctionEntries(
    @e001 bigint = NULL,
    @s001 uniqueidentifier = NULL,
    @l001 image = NULL,
    @e002 bigint = NULL,
    @s002 uniqueidentifier = NULL,
    @l002 image = NULL,
    @e003 bigint = NULL,
    @s003 uniqueidentifier = NULL,
    @l003 image = NULL,
    @e004 bigint = NULL,
    @s004 uniqueidentifier = NULL,
    @l004 image = NULL,
    @e005 bigint = NULL,
    @s005 uniqueidentifier = NULL,
    @l005 image = NULL,
    @e006 bigint = NULL,
    @s006 uniqueidentifier = NULL,
    @l006 image = NULL,
    @e007 bigint = NULL,
    @s007 uniqueidentifier = NULL,
    @l007 image = NULL,
    @e008 bigint = NULL,
    @s008 uniqueidentifier = NULL,
    @l008 image = NULL,
    @e009 bigint = NULL,
    @s009 uniqueidentifier = NULL,
    @l009 image = NULL,
    @e010 bigint = NULL,
    @s010 uniqueidentifier = NULL,
    @l010 image = NULL,
    @e011 bigint = NULL,
    @s011 uniqueidentifier = NULL,
    @l011 image = NULL,
    @e012 bigint = NULL,
    @s012 uniqueidentifier = NULL,
    @l012 image = NULL,
    @e013 bigint = NULL,
    @s013 uniqueidentifier = NULL,
    @l013 image = NULL,
    @e014 bigint = NULL,
    @s014 uniqueidentifier = NULL,
    @l014 image = NULL,
    @e015 bigint = NULL,
    @s015 uniqueidentifier = NULL,
    @l015 image = NULL,
    @e016 bigint = NULL,
    @s016 uniqueidentifier = NULL,
    @l016 image = NULL,
    @e017 bigint = NULL,
    @s017 uniqueidentifier = NULL,
    @l017 image = NULL,
)
```

[MS-WSSDLIM] — v20120630
Windows SharePoint Services: Content Database Document and List Item Management Communications Protocol Specification

Copyright © 2012 Microsoft Corporation.

Release: July 16, 2012
@e018 bigint = NULL,
@e018 uniqueidentifier = NULL,
@e018 image = NULL,
@e019 bigint = NULL,
@e019 uniqueidentifier = NULL,
@e019 image = NULL,
@e020 bigint = NULL,
@e020 uniqueidentifier = NULL,
@e020 image = NULL,
@e021 bigint = NULL,
@e021 uniqueidentifier = NULL,
@e021 image = NULL,
@e022 bigint = NULL,
@e022 uniqueidentifier = NULL,
@e022 image = NULL,
@e023 bigint = NULL,
@e023 uniqueidentifier = NULL,
@e023 image = NULL,
@e024 bigint = NULL,
@e024 uniqueidentifier = NULL,
@e024 image = NULL,
@e025 bigint = NULL,
@e025 uniqueidentifier = NULL,
@e025 image = NULL,
@e026 bigint = NULL,
@e026 uniqueidentifier = NULL,
@e026 image = NULL,
@e027 bigint = NULL,
@e027 uniqueidentifier = NULL,
@e027 image = NULL,
@e028 bigint = NULL,
@e028 uniqueidentifier = NULL,
@e028 image = NULL,
image = NULL,
bigint = NULL,
uniqueidentifier = NULL,
image = NULL,
bigint = NULL,
uniqueidentifier = NULL,
image = NULL,
bigint = NULL,
uniqueidentifier = NULL,
image = NULL,
bigint = NULL,
uniqueidentifier = NULL,
image = NULL,
bigint = NULL,
uniqueidentifier = NULL,
image = NULL,
bigint = NULL,
uniqueidentifier = NULL,
image = NULL,
bigint = NULL,
uniqueidentifier = NULL,
image = NULL,
bigint = NULL,
uniqueidentifier = NULL,
uniqueidentifier = NULL,
image = NULL,
bigint = NULL,
uniqueidentifier = NULL,
image = NULL,
bigint = NULL,
uniqueidentifier = NULL,
image = NULL,
bigint = NULL,
uniqueidentifier = NULL,
image = NULL,
bigint = NULL,
uniqueidentifier = NULL,
image = NULL,
bigint = NULL,
uniqueidentifier = NULL,
image = NULL,
bigint = NULL,
uniqueidentifier = NULL,
image = NULL,
bigint = NULL,
uniqueidentifier = NULL,
image = NULL,
bigint = NULL,
uniqueidentifier = NULL,
image = NULL,
bigint = NULL,
uniqueidentifier = NULL,
image = NULL,
@e077 bigint = NULL,
@s077 uniqueidentifier = NULL,
@l077 image = NULL,
@e078 bigint = NULL,
@s078 uniqueidentifier = NULL,
@l078 image = NULL,
@e079 bigint = NULL,
@s079 uniqueidentifier = NULL,
@l079 image = NULL,
@e080 bigint = NULL,
@s080 uniqueidentifier = NULL,
@l080 image = NULL,
@e081 bigint = NULL,
@s081 uniqueidentifier = NULL,
@l081 image = NULL,
@e082 bigint = NULL,
@s082 uniqueidentifier = NULL,
@l082 image = NULL,
@e083 bigint = NULL,
@s083 uniqueidentifier = NULL,
@l083 image = NULL,
@e084 bigint = NULL,
@s084 uniqueidentifier = NULL,
@l084 image = NULL,
@e085 bigint = NULL,
@s085 uniqueidentifier = NULL,
@l085 image = NULL,
@e086 bigint = NULL,
@s086 uniqueidentifier = NULL,
@l086 image = NULL,
@e087 bigint = NULL,
@s087 uniqueidentifier = NULL,
@l087 image = NULL,
@e088 bigint = NULL,
@s088 uniqueidentifier = NULL,
@l088 image = NULL,
@e089 bigint = NULL,
@s089 uniqueidentifier = NULL,
@l089 image = NULL,
@e090 bigint = NULL,
@s090 uniqueidentifier = NULL,
@l090 image = NULL,
@e091 bigint = NULL,
@s091 uniqueidentifier = NULL,
@l091 image = NULL,
@e092 bigint = NULL,
@s092 uniqueidentifier = NULL,
@l092 image = NULL,
@e093 bigint = NULL,
@s093 uniqueidentifier = NULL,
@l093 image = NULL,
@e094 bigint = NULL,
@s094 uniqueidentifier = NULL,
@l094 image = NULL,
@e095 bigint = NULL,
@s095 uniqueidentifier = NULL,
@l095 image = NULL,
@e096 bigint = NULL,
@s096 uniqueidentifier = NULL,
@l096 image = NULL,
@e097 bigint = NULL,
@e097 uniqueidentifier = NULL,
@l097 image = NULL,
@e098 bigint = NULL,
@e098 uniqueidentifier = NULL,
@l098 image = NULL,
@e099 bigint = NULL,
@e099 uniqueidentifier = NULL,
@l099 image = NULL,
@e100 bigint = NULL,
@e100 uniqueidentifier = NULL,
@l100 image = NULL,
@e101 bigint = NULL,
@e101 uniqueidentifier = NULL,
@l101 image = NULL,
@e102 bigint = NULL,
@e102 uniqueidentifier = NULL,
@l102 image = NULL,
@e103 bigint = NULL,
@e103 uniqueidentifier = NULL,
@l103 image = NULL,
@e104 bigint = NULL,
@e104 uniqueidentifier = NULL,
@l104 image = NULL,
@e105 bigint = NULL,
@e105 uniqueidentifier = NULL,
@l105 image = NULL,
@e106 bigint = NULL,
@e106 uniqueidentifier = NULL,
@l106 image = NULL,
@e107 bigint = NULL,
@e107 uniqueidentifier = NULL,
@l107 image = NULL,
@e108 bigint = NULL,
@e108 uniqueidentifier = NULL,
@l108 image = NULL,
@e109 bigint = NULL,
@e109 uniqueidentifier = NULL,
@l109 image = NULL,
@e110 bigint = NULL,
@e110 uniqueidentifier = NULL,
@l110 image = NULL,
@e111 bigint = NULL,
@e111 uniqueidentifier = NULL,
@l111 image = NULL,
@e112 bigint = NULL,
@e112 uniqueidentifier = NULL,
@l112 image = NULL,
@e113 bigint = NULL,
@e113 uniqueidentifier = NULL,
@l113 image = NULL,
@e114 bigint = NULL,
@e114 uniqueidentifier = NULL,
@l114 image = NULL,
@e115 bigint = NULL,
@e115 uniqueidentifier = NULL,
@l115 image = NULL,
@e116 bigint = NULL,
| @s116 | uniqueidentifier = NULL, |
| @l116 | image = NULL, |
| @e117 | bigint = NULL, |
| @s117 | uniqueidentifier = NULL, |
| @l117 | image = NULL, |
| @e118 | bigint = NULL, |
| @s118 | uniqueidentifier = NULL, |
| @l118 | image = NULL, |
| @e119 | bigint = NULL, |
| @s119 | uniqueidentifier = NULL, |
| @l119 | image = NULL, |
| @e120 | bigint = NULL, |
| @s120 | uniqueidentifier = NULL, |
| @l120 | image = NULL, |
| @e121 | bigint = NULL, |
| @s121 | uniqueidentifier = NULL, |
| @l121 | image = NULL, |
| @e122 | bigint = NULL, |
| @s122 | uniqueidentifier = NULL, |
| @l122 | image = NULL, |
| @e123 | bigint = NULL, |
| @s123 | uniqueidentifier = NULL, |
| @l123 | image = NULL, |
| @e124 | bigint = NULL, |
| @s124 | uniqueidentifier = NULL, |
| @l124 | image = NULL, |
| @e125 | bigint = NULL, |
| @s125 | uniqueidentifier = NULL, |
| @l125 | image = NULL, |
| @e126 | bigint = NULL, |
| @s126 | uniqueidentifier = NULL, |
| @l126 | image = NULL, |
| @e127 | bigint = NULL, |
| @s127 | uniqueidentifier = NULL, |
| @l127 | image = NULL, |
| @e128 | bigint = NULL, |
| @s128 | uniqueidentifier = NULL, |
| @l128 | image = NULL, |
| @e129 | bigint = NULL, |
| @s129 | uniqueidentifier = NULL, |
| @l129 | image = NULL, |
| @e130 | bigint = NULL, |
| @s130 | uniqueidentifier = NULL, |
| @l130 | image = NULL, |
| @e131 | bigint = NULL, |
| @s131 | uniqueidentifier = NULL, |
| @l131 | image = NULL, |
| @e132 | bigint = NULL, |
| @s132 | uniqueidentifier = NULL, |
| @l132 | image = NULL, |
| @e133 | bigint = NULL, |
| @s133 | uniqueidentifier = NULL, |
| @l133 | image = NULL, |
| @e134 | bigint = NULL, |
| @s134 | uniqueidentifier = NULL, |
| @l134 | image = NULL, |
| @e135 | bigint = NULL, |
| @s135 | uniqueidentifier = NULL, |
| @l135 | image = NULL,
@e136   bigint = NULL,
@e136   uniqueidentifier = NULL,
@l136   image = NULL,
@e137   bigint = NULL,
@e137   uniqueidentifier = NULL,
@l137   image = NULL,
@e138   bigint = NULL,
@e138   uniqueidentifier = NULL,
@l138   image = NULL,
@e139   bigint = NULL,
@e139   uniqueidentifier = NULL,
@l139   image = NULL,
@e140   bigint = NULL,
@e140   uniqueidentifier = NULL,
@l140   image = NULL,
@e141   bigint = NULL,
@e141   uniqueidentifier = NULL,
@l141   image = NULL,
@e142   bigint = NULL,
@e142   uniqueidentifier = NULL,
@l142   image = NULL,
@e143   bigint = NULL,
@e143   uniqueidentifier = NULL,
@l143   image = NULL,
@e144   bigint = NULL,
@e144   uniqueidentifier = NULL,
@l144   image = NULL,
@e145   bigint = NULL,
@e145   uniqueidentifier = NULL,
@l145   image = NULL,
@e146   bigint = NULL,
@e146   uniqueidentifier = NULL,
@l146   image = NULL,
@e147   bigint = NULL,
@e147   uniqueidentifier = NULL,
@l147   image = NULL,
@e148   bigint = NULL,
@e148   uniqueidentifier = NULL,
@l148   image = NULL,
@e149   bigint = NULL,
@e149   uniqueidentifier = NULL,
@l149   image = NULL,
@e150   bigint = NULL,
@e150   uniqueidentifier = NULL,
@l150   image = NULL,
@e151   bigint = NULL,
@e151   uniqueidentifier = NULL,
@l151   image = NULL,
@e152   bigint = NULL,
@e152   uniqueidentifier = NULL,
@l152   image = NULL,
@e153   bigint = NULL,
@e153   uniqueidentifier = NULL,
@l153   image = NULL,
@e154   bigint = NULL,
@e154   uniqueidentifier = NULL,
@l154   image = NULL,
@e155   bigint = NULL,
@e155   uniqueidentifier = NULL,
@s175  uniqueidentifier = NULL,
@s175  image = NULL,
@e176  bigint = NULL,
@s176  uniqueidentifier = NULL,
@s176  image = NULL,
@e177  bigint = NULL,
@s177  uniqueidentifier = NULL,
@1177  image = NULL,
@e178  bigint = NULL,
@s178  uniqueidentifier = NULL,
@s178  image = NULL,
@e179  bigint = NULL,
@s179  uniqueidentifier = NULL,
@1179  image = NULL,
@e180  bigint = NULL,
@s180  uniqueidentifier = NULL,
@1180  image = NULL,
@e181  bigint = NULL,
@s181  uniqueidentifier = NULL,
@1181  image = NULL,
@e182  bigint = NULL,
@s182  uniqueidentifier = NULL,
@1182  image = NULL,
@e183  bigint = NULL,
@s183  uniqueidentifier = NULL,
@1183  image = NULL,
@e184  bigint = NULL,
@s184  uniqueidentifier = NULL,
@1184  image = NULL,
@e185  bigint = NULL,
@s185  uniqueidentifier = NULL,
@1185  image = NULL,
@e186  bigint = NULL,
@s186  uniqueidentifier = NULL,
@1186  image = NULL,
@e187  bigint = NULL,
@s187  uniqueidentifier = NULL,
@1187  image = NULL,
@e188  bigint = NULL,
@s188  uniqueidentifier = NULL,
@1188  image = NULL,
@e189  bigint = NULL,
@s189  uniqueidentifier = NULL,
@1189  image = NULL,
@e190  bigint = NULL,
@s190  uniqueidentifier = NULL,
@1190  image = NULL,
@e191  bigint = NULL,
@s191  uniqueidentifier = NULL,
@1191  image = NULL,
@e192  bigint = NULL,
@s192  uniqueidentifier = NULL,
@1192  image = NULL,
@e193  bigint = NULL,
@s193  uniqueidentifier = NULL,
@1193  image = NULL,
@e194  bigint = NULL,
@s194  uniqueidentifier = NULL,
@1194  image = NULL,
@e195  bigint = NULL,  
@s195  uniqueidentifier = NULL,  
@l195  image = NULL,  
@e196  bigint = NULL,  
@s196  uniqueidentifier = NULL,  
@l196  image = NULL,  
@e197  bigint = NULL,  
@s197  uniqueidentifier = NULL,  
@l197  image = NULL,  
@e198  bigint = NULL,  
@s198  uniqueidentifier = NULL,  
@l198  image = NULL,  
@e199  bigint = NULL,  
@s199  uniqueidentifier = NULL,  
@l199  image = NULL,  
@e200  bigint = NULL,  
@s200  uniqueidentifier = NULL,  
@l200  image = NULL,  
@e201  bigint = NULL,  
@s201  uniqueidentifier = NULL,  
@l201  image = NULL,  
@e202  bigint = NULL,  
@s202  uniqueidentifier = NULL,  
@l202  image = NULL,  
@e203  bigint = NULL,  
@s203  uniqueidentifier = NULL,  
@l203  image = NULL,  
@e204  bigint = NULL,  
@s204  uniqueidentifier = NULL,  
@l204  image = NULL,  
@e205  bigint = NULL,  
@s205  uniqueidentifier = NULL,  
@l205  image = NULL,  
@e206  bigint = NULL,  
@s206  uniqueidentifier = NULL,  
@l206  image = NULL,  
@e207  bigint = NULL,  
@s207  uniqueidentifier = NULL,  
@l207  image = NULL,  
@e208  bigint = NULL,  
@s208  uniqueidentifier = NULL,  
@l208  image = NULL,  
@e209  bigint = NULL,  
@s209  uniqueidentifier = NULL,  
@l209  image = NULL,  
@e210  bigint = NULL,  
@s210  uniqueidentifier = NULL,  
@l210  image = NULL,  
@e211  bigint = NULL,  
@s211  uniqueidentifier = NULL,  
@l211  image = NULL,  
@e212  bigint = NULL,  
@s212  uniqueidentifier = NULL,  
@l212  image = NULL,  
@e213  bigint = NULL,  
@s213  uniqueidentifier = NULL,  
@l213  image = NULL,  
@e214  bigint = NULL,  
@s214  uniqueidentifier = NULL,
uniqueidentifier = NULL,
image = NULL,
b bigint = NULL,
uniqueidentifier = NULL,
image = NULL,
bigint = NULL,
uniqueidentifier = NULL,
image = NULL,
bigint = NULL,
uniqueidentifier = NULL,
image = NULL,
@ennn: The event identifier corresponding to a unique Event for which Subscription data has to be inserted.

@snnn: The Subscription IDENTIFIER for which Subscription data has to be inserted. If corresponding @ennn is not NULL then @snnn MUST not be NULL.

@l1nnn: A 64 bit field that holds the permissions for the User for the lookup field.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST NOT return any Result Sets.

3.1.4.44 proc_InsertItemIntoNameValuePair

The proc_InsertItemIntoNameValuePair Stored Procedure is called to insert Indexed Fields and their values for the specified List Item. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_InsertItemIntoNameValuePair(
    @SiteId uniqueidentifier,
    @WebId uniqueidentifier,
    @ListId uniqueidentifier,
    @ItemId int,
    @Level tinyint = 1,
    @FieldId1 uniqueidentifier = NULL,
    @FieldValue1 sql_variant = NULL,
    @FieldId2 uniqueidentifier = NULL,
    @FieldValue2 sql_variant = NULL,
    @FieldId3 uniqueidentifier = NULL,
    @FieldValue3 sql_variant = NULL,
    @FieldId4 uniqueidentifier = NULL,
    @FieldValue4 sql_variant = NULL,
    @FieldId5 uniqueidentifier = NULL,
    @FieldValue5 sql_variant = NULL,
    @FieldId6 uniqueidentifier = NULL,
    @FieldValue6 sql_variant = NULL,
    @FieldId7 uniqueidentifier = NULL,
    @FieldValue7 sql_variant = NULL,
    @FieldId8 uniqueidentifier = NULL,
    @FieldValue8 sql_variant = NULL,
    @FieldId9 uniqueidentifier = NULL,
    @FieldValue9 sql_variant = NULL,
    @FieldId10 uniqueidentifier = NULL,
    @FieldValue10 sql_variant = NULL,
    @SelectFromUserData bit = 0
);
@SiteId: The Site Collection Identifier of the Site Collection which contains the specified List Item.

@WebId: The Site Identifier of the Site which contains the specified List Item.

@ListId: The List Identifier of the List which contains the specified List Item.

@ItemId: The Identifier of the specified List Item in the List.

@Level: The Publishing Level. The default value is 1.

@FieldId#: The GUID of the Indexed Fields. There are ten FieldID parameters numbered from 1 to 10. The default values are NULL.

@FieldValue#: The value of the Indexed Fields. There are ten FieldValue parameters numbered from 1 to 10. The default values are NULL.

@SelectFromUserData: An input parameter. If it is set to 1 and @FieldId# is not NULL and @FieldId# is one of the Field Identifier values in the following table, then associated @FieldValue# is replaced with the corresponding column of the row in the AllUserData specified by @ListId, @ItemId and @Level. The default value is 0. This parameter MUST NOT be NULL.

<table>
<thead>
<tr>
<th>Field Value(Column in the AllUserData Table)</th>
<th>Field Id</th>
</tr>
</thead>
<tbody>
<tr>
<td>tp_Author</td>
<td>{1df5e554-ec7e-46a6-901d-d85a3881cb18}</td>
</tr>
<tr>
<td>tp_Editor</td>
<td>{d31655d1-1d5b-4511-95a1-7a09e9b75bf2}</td>
</tr>
<tr>
<td>tp_Created</td>
<td>{8c06beca-0777-48f7-91c7-6da68bc07b69}</td>
</tr>
<tr>
<td>tp_Modified</td>
<td>{28cf69c5-fa48-462a-b5cd-27b6f9d2bd5f}</td>
</tr>
<tr>
<td>tp_HasCopyDestinations</td>
<td>{26d0756c-986a-48a7-af35-bf18ab85ff4a}</td>
</tr>
<tr>
<td>tp_UIVersion</td>
<td>{7841bf41-43d0-4434-9f50-a673baef7631}</td>
</tr>
<tr>
<td>tp_ContentTypeId</td>
<td>{03e45e84-1992-4d42-9116-26f756012634}</td>
</tr>
<tr>
<td>tp_CheckoutUserId</td>
<td>{3881510a-4e4a-4ee8-b102-8ee8e2d0dd4b}</td>
</tr>
</tbody>
</table>

Return Code Values: An integer which MUST be listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful execution.</td>
</tr>
<tr>
<td>87</td>
<td>The input parameters are incorrect.</td>
</tr>
</tbody>
</table>

Result Sets: MUST NOT return any Result Sets.

3.1.4.45   proc_InsertJunction

The proc_InsertJunction Stored Procedure is called to add a value to the set of values of a Multi-valued lookup field of a specified List Item in a List. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_InsertJunction(
    @SiteId uniqueidentifier,
```
@SiteId: The Site Collection Identifier of the Site Collection which contains the specified List which contains the specified multivalued lookup field.

@DirName: The Directory Name of the specified List Item.

@LeafName: The Leaf Name of the specified List Item.

@FieldId: The Field Identifier of the specified multivalued lookup field.

@Id: The row identifier of a List Item in the List being looked up by the specified multivalued lookup field.

@Ordinal: It MUST be a 0-based ordinal of the row which contains the column corresponding to the specified lookup field. Additional rows are used when a List has more user-defined columns of one or more data types than can fit in a single row of this view.

@Level: A Publishing Level value specifying the publishing status of this List Item.

@UIVersion: A User Interface (UI) Version number associated with the List Item.

@IsCurrentVersion: A bit flag specifying whether the specified row belongs to the Current Version of the List Item. This parameter SHOULD be set to its default value of 1 to appear in the UserDataJunctions view.

@CalculatedVersion: This Field keeps all live/nonhistorical rows for a List Item together. This SHOULD equal 0 to appear in the UserDataJunctions view, as the UserDataJunctions view only contains non-historical List Items.

@DeleteTransactionId: A parameter which identifies the delete transaction identifier which encapsulated the actual delete operation. This is used so that multiple or hierarchical operations MAY be performed by the caller. This parameter SHOULD be set to its default value of 0x to appear in the UserDataJunctions view.

Return Code Values: An integer which MUST be listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful execution.</td>
</tr>
<tr>
<td>-4</td>
<td>Insertion Error. @SiteId, @DeleteTransactionId, @IsCurrentVersion, @FieldId, @CalculatedVersion, @Level, @Ordinal do not form a unique entry in AllUserDataJunctions.</td>
</tr>
</tbody>
</table>

Result Sets: MUST NOT return any Result Sets.
3.1.4.46  proc_ListDocsInCategory

The proc_ListDocsInCategory Stored Procedure is called to retrieve the Documents associated with a Category. The T-SQL syntax for the Stored Procedure is as follows:

```
PROCEDURE proc_ListDocsInCategory(
    @SiteId            uniqueidentifier,
    @WebDirName        nvarchar(256),
    @WebLeafName       nvarchar(128),
    @Category          nvarchar(128),
    @FullMetaInfo      bit
);
```

@SiteId: The Site Collection Identifier of the Site Collection.

@WebDirName: The Directory Name of the Site containing the Documents associated with the Category. This parameter MUST be the Directory Name for a Site that exists in the Site Collection.

@WebLeafName: The Leaf Name of the Site containing the Documents associated with the Category. This parameter MUST be the Leaf Name for a Site that exists in the Site Collection.

@Category: Category for which Documents are retrieved.

@FullMetaInfo: Bit flag indicating whether to return the Metadict of Documents associated with the Category. This bit flag MUST have a value of 0 or 1. A value of 0 indicates that a DocsCategory result set MUST be returned. A value of 1 indicates that a DocsCategoryMetaInfo result set MUST be returned.

Return Values: proc_ListDocsInCategory returns an integer Return Code which MUST be listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>The Stored Procedure has finished execution.</td>
</tr>
<tr>
<td>3</td>
<td>The Site or Site Collection does not exist.</td>
</tr>
</tbody>
</table>

Result Sets: MUST return a Docs Category result set or a Docs Category Meta Info result set.

3.1.4.46.1  Docs Category Result Set

Docs Category returns Document properties for Documents associated with the Category. The Docs Category Result Set will be returned if the @FullMetaInfo parameter is 0. The number of rows is the number of Documents in the Site associated with the Category. The T-SQL syntax for the result set is as follows:

```
DirName       nvarchar(256),
LeafName      nvarchar(128);
```

DirName: Contains the Directory Name of the Document associated with the Category.

LeafName: Contains the Leaf Name of the Document associated with the Category.
3.1.4.46.2 DocsCategoryMetaInfo Result Set

Docs Category Meta Info returns Document properties and metadata for Documents associated with the Category. Docs Category Meta Info Result Set will be returned if the @FullMetaInfo parameter is 1. The number of rows is the number of Documents in the Site associated with the Category. The T-SQL syntax for the result set is as follows:

```
DirsName          nvarchar(256),
LeafName          nvarchar(128),
TimeLastModified  datetime,
MetaInfo          image;
```

**DirName**: Contains the Directory Name of the Document associated with the Category.

**LeafName**: Contains the Leaf Name of the Document associated with the Category.

**TimeLastModified**: Contains the date and time when the Document associated with the Category was last modified.

**MetaInfo**: Contains the Metadict of the Document associated with the Category.

3.1.4.47 proc_ListThemeFiles

The proc_ListThemeFiles Stored Procedure is called to retrieve information about all related files for a given Theme. The T-SQL syntax for the Stored Procedure is as follows:

```
PROCEDURE proc_ListThemeFiles(
    @SiteId         uniqueidentifier,
    @WebUrl         nvarchar(260),
    @ThemeName      nvarchar(128)
);
```

**@SiteId**: The Site Collection Identifier of the Site Collection which contains the Site specified by the @WebUrl parameter.

**@WebUrl**: The URL in Store-Relative Form of the Site that contains the Theme specified by the @ThemeName Stored Procedure parameter. Specifying NULL for this parameter will retrieve the Theme files for the Top-level site of the Site Collection specified by the @SiteId parameter.

**@ThemeName**: The Unicode string name of the Theme to retrieve the files.

**Return Code Values**: An integer which MUST be 0.

**Result Sets**: MUST return one result set.

3.1.4.47.1 Theme Files Information Result Set

This Result Set contains information about the Theme related files associated with the Theme specified by parameter @ThemeName, for a Site specified by parameter @WebUrl that belongs to the Site Collection specified by parameter @SiteId. There MUST be one row for each such Theme related file. The T-SQL syntax for the result set is as follows:

```
{FileURL}          nvarchar(385),
Content            image,
HasStream          int,
```
SetupPathVersion tinyint,
SetupPath nvarchar(255);

{FileURL}: The URL of a file in Store-Relative Form for the Theme.

Content: For uncustomized theme files this MUST be NULL. For files that are not uncustomized, this
MUST be the content of the theme file as a binary image.

HasStream: MUST be "1" if the theme file is uncustomized. Otherwise, MUST be zero ("0") if the
theme file is customized (1).

SetupPathVersion: This MUST be one of the following with regards to the theme file:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NULL</td>
<td>This file has never been uncustomized.</td>
</tr>
<tr>
<td>&quot;2&quot;</td>
<td>The path of the file is relative to the installation location of wss2 on the front-end Web server. For example, Program Files\Common Files\Microsoft Shared\Web Server Extensions\60.</td>
</tr>
<tr>
<td>&quot;3&quot;</td>
<td>The path of the file is relative to the installation location of wss3 on the front-end Web server. For example, Program Files\Common Files\Microsoft Shared\Web Server Extensions\12\template.</td>
</tr>
</tbody>
</table>

SetupPath: For a theme file that has ever been uncustomized, this MUST be the path fragment
relative to the path returned by the SetupPathVersion column value. Taken together, the
SetupPathVersion and SetupPath columns determine where theme files are located on the front-
end Web server's file system. For example, if SetupPathVersion is "3" and SetupPath is
"themes\lichen\LICHEN.INF", the path is "Program Files\Common Files\Microsoft Shared\Web Server Extensions\12\template\themes\lichen\LICHEN.INF." This MUST be NULL if the theme file
has never been uncustomized.

3.1.4.48 proc_ListThemes

The proc_ListThemes Stored Procedure is called to return information about a Theme that is
contained within a Site. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_ListThemes(
    @SiteId uniqueidentifier,
    @WebUrl nvarchar(260)
);
```

@SiteId: The Site Collection Identifier of the Site Collection which contains the Site specified by the
@WebUrl parameter.

@WebUrl: The URL in Store-Relative Form of the Site whose Theme information will be retrieved.
Specifying NULL for this parameter will retrieve the Theme information for the Top-level site of the
Site Collection specified by the @SiteId parameter.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST return one result set.
3.1.4.48.1 Theme Information Result Set

This Result Set contains information about the Theme contained within the Site specified by parameter @WebUrl that belongs to the Site Collection specified by parameter @SiteId. For combinations of the @WebUrl and @SiteId parameters that define an existing Site, one row MUST be returned. Otherwise, an empty Result Set MUST be returned. The T-SQL syntax for the result set is as follows:

```
LeafName            nvarchar(128),
SetupPathVersion    tinyint,
SetupPath           nvarchar(255),
SetupPathUser       nvarchar(255),
Content             image;
```

**LeafName:** The Theme name.

**SetupPathVersion:** This MUST be one of the following:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>The path to the inf file that describes the theme returned by the Theme Information Result Set. SetupPath column is relative to the installation location of wss2 on the front-end Web server. For example, Program Files\Common Files\Microsoft Shared\Web Server Extensions\60.</td>
</tr>
<tr>
<td>3</td>
<td>The path to the Theme INF File returned by the Theme Information Result Set SetupPath column is relative to the installation location of wss3 on the front-end Web server. For example, Program Files\Common Files\Microsoft Shared\Web Server Extensions\12\template.</td>
</tr>
</tbody>
</table>

**SetupPath:** This MUST be the path fragment of the Theme INF File that describes the Theme. Taken together, the SetupPathVersion and SetupPath columns determine where the Theme INF File is located on the front-end Web server's file system. For example, if the SetupPathVersion is 3 and the SetupPath is themes\Lichen\LICHEN.INF, then the path to the Theme INF File is "Program Files\Common Files\Microsoft Shared\Web Server Extensions\12\template\themes\Lichen\LICHEN.INF".

**SetupPathUser:** This contains the login name of the user that created the theme.

**Content:** If the theme INF File is uncustomized, this MUST be NULL. If the theme INF File is not uncustomized, this MUST be the content of the theme INF File.

3.1.4.49 proc_LoadTheme

The proc_LoadTheme Stored Procedure is called to return data about certain files related to a given Theme. The T-SQL syntax for the Stored Procedure is as follows:

```
PROCEDURE proc_LoadTheme(
    @WebSiteId        uniqueidentifier,
    @ThemesDir        nvarchar(256),
    @ThemeName        nvarchar(128),
    @GraphicCSS       nvarchar(128),
    @ColorCSS         nvarchar(128),
    @ExtCSS           nvarchar(128),
    @NeedThemesInf    bit
);```
@WebSiteId: The Site Collection Identifier of the Site Collection to retrieve the Theme data from.

@ThemesDir: The URL in Store-Relative Form of the Theme. This Stored Procedure parameter MUST be one of the following:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>_themes</td>
<td>Retrieves the Theme data for the Top-level site of the Site Collection specified by the @WebSiteId Stored Procedure parameter.</td>
</tr>
<tr>
<td>[Site]/_themes</td>
<td>Retrieves the Theme data for a Site that belongs to the Site Collection specified by the @WebSiteId Stored Procedure parameter. For example, if “Contoso” is the name of a Site that belongs to the Site Collection, then to retrieve the Theme data for the “Contoso” Site, the @ThemesDir parameter would be: ‘Contoso/_themes’</td>
</tr>
</tbody>
</table>

@ThemeName: Unicode string name of the Theme.

@GraphicCSS: The Unicode string file name of a Cascading Style Sheet (CSS) that defines how text is displayed in a user interface for a Theme in the navigation bars. For example, this Cascading Style Sheet (CSS) MAY define the images used for displaying Hyperlinks, the font-family, font-size, font-style, font-weight, font color, text-align of text to be displayed in the navigation bars. MUST be one of the following:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>graph0.css</td>
<td>The Cascading Style Sheet (CSS) defined for Theme text that is not active in a Top Link Bar. For example, a link in is not active if the end user has not interacted with it by clicking on the link with a mouse.</td>
</tr>
<tr>
<td>graph1.css</td>
<td>The Cascading Style Sheet (CSS) defined for Theme text that is active in a Top Link Bar. For example, a link in a Theme is said to be active when the end user clicks on the link with a mouse.</td>
</tr>
</tbody>
</table>

@ColorCSS: The Unicode string file name of a Cascading Style Sheet (CSS) that defines the color for text, Hyperlinks, and background images in a user interface for a Theme. MUST be one of the following:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>color0.css</td>
<td>The Cascading Style Sheet (CSS) for the normal color set for the Theme. The normal color set is comprised of colors that are traditionally used and more aptly supported.</td>
</tr>
<tr>
<td>color1.css</td>
<td>The Cascading Style Sheet (CSS) for the vivid color set for the Theme. The vivid color set expands the normal color set with a brighter set of colors.</td>
</tr>
</tbody>
</table>

@ExtCSS: The masked Unicode string file name that matches a set of Cascading Style Sheet (CSS) files whose location or contents will be returned. Here, "masked" refers to the fact that @ExtCSS MUST be specified exactly using the following pattern:

'_%extension.css'
This Stored Procedure will interpret this and return any Cascading Style Sheet (CSS) whose file name suffix is 'extension.css' and where '%' can be replaced with any character or characters. For example, the Cascading Style Sheet (CSS) file name of 'contosoextension.css' would be returned. Those Cascading Style Sheet (CSS) definitions that cannot be defined in the Cascading Style Sheet (CSS) files provided by the @ColorCSS and @GraphicCSS Stored Procedure parameters MUST be defined in the Cascading Style Sheet (CSS) files that match this pattern.

@NeedThemesInf: Specifies if this Stored Procedure has to return information about the Theme INF File. MUST be one of the following values:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>MUST NOT return information about the Theme INF File.</td>
</tr>
<tr>
<td>1</td>
<td>MUST return information about the Theme INF File.</td>
</tr>
</tbody>
</table>

Return Code Values: An integer which MUST be listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>The Stored Procedure parameter @NeedThemesInf has value 0 or the Stored Procedure parameter @NeedThemesInf has value 1 and the corresponding Theme INF File was found.</td>
</tr>
<tr>
<td>1</td>
<td>The Stored Procedure parameter @NeedThemesInf has value 1 which indicated that the Theme INF File was specifically requested, but it was not found.</td>
</tr>
</tbody>
</table>

Result Sets: MUST return two Result Sets as follows:

3.1.4.49.1 Theme Files Information Result Set

This Result Set contains information about the following Theme files:

1. 'theme.css'
2. 'custom.css'
3. The Unicode string given by the parameter @GraphicCSS
4. The Unicode string given by the parameter @ColorCSS
5. The Unicode string that matches the pattern given by the parameter @ExtCSS.
6. The Unicode string given by concatenation of the value from the parameter @ThemeName and extension '.utf8'.
7. The Unicode string given by concatenation of the value from the parameter @ThemeName and extension '.inf' if the file with the extension '.utf8' was not found.

Information about files that are found according to the preceding criteria will be returned as a row per qualifying file in the result set. Note that more than one row MAY be returned for Cascading Style Sheet (CSS) file names that match the criteria specified by the @ExtCSS parameter. The T-SQL syntax for the result set is as follows:

```sql
LEAFNAME nvarchar(128),
SETUPPATHVERSION tinyint,
SETUPPATH nvarchar(255),
```

[MS-WSSDLIM] — v20120630
Windows SharePoint Services: Content Database Document and List Item Management Communications Protocol Specification

Copyright © 2012 Microsoft Corporation.

Release: July 16, 2012
LeafName: The Theme name.

SetupPathVersion: This MUST be one of the following:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>The path to the Theme file is relative to the installation location of wss2 on the front-end Web server. For example, Program Files\Common Files\Microsoft Shared\Web Server Extensions\60.</td>
</tr>
<tr>
<td>3</td>
<td>The path to the Theme file is relative to the installation location of wss3 on the front-end Web server. For example, Program Files\Common Files\Microsoft Shared\Web Server Extensions\12\template.</td>
</tr>
</tbody>
</table>

SetupPath: This MUST be the path fragment of the Theme file. Taken together, the SetupPathVersion and SetupPath columns determine where the Theme file is located on the front-end Web server's file system. For example, if the SetupPathVersion is 3 and the SetupPath is themes\Lichen\LICHEN.INF, then the path would be 'Program Files\Common Files\Microsoft Shared\Web Server Extensions\12\template\themes\Lichen\LICHEN.INF'.

SetupPathUser: This contains the Login Name of the User that created the Theme.

Content: If the theme file is uncustomized, this MUST be NULL. If the theme file is not uncustomized, this MUST be the content of the theme file.

3.1.4.49.2 Theme INF File Info Result Set

This Result Set contains information about the Theme INF File. If the Theme INF File can be found, then one row MUST be returned. If the Theme INF File cannot be found, then no rows MUST be returned. The name of the Theme INF File MUST be the Unicode string obtained by concatenating the value from the parameter @ThemeName and the extension '.inf'. The T-SQL syntax for the result set is as follows:

```sql
LeafName          nvarchar(128),
SetupPathVersion  tinyint,
SetupPath         nvarchar(255),
SetupPathUser     nvarchar(255),
Content           image
```

The semantic of these parameters is the same as for those in the previously defined "Theme Files Information Result Set."

3.1.4.50 proc_LogChange

The proc_LogChange Stored Procedure is called to store the information about an Event that is either triggered by the user or triggered by the system. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_LogChange(
    @SiteId                      uniqueidentifier,
    @WebId                       uniqueidentifier,
    @ListId                      uniqueidentifier,
```
@SiteId: This value is a Change Log SiteId (Section 2.2.1.11).
@WebId: This value is a Change Log WebId (Section 2.2.1.12).
@ListId: This value is a Change Log ListId (Section 2.2.1.1).
@ItemId: This value is a Change Log ItemId (Section 2.2.1.2).
@DocId: This value is a Change Log DocId (Section 2.2.1.2).
@Guid0: This value is a Change Log Guid0 (Section 2.2.1.4).
@Int0: This value is a Change Log Int0 (Section 2.2.1.4).
@FullUrl: This value is a Change Log ItemFullUrl (Section 2.2.1.7).
@EventType: An integer that represents the Event Type Flags.
@ObjectType: An integer that represents the Event Object Type Flags.
@TimeLastModifiedIncoming: This value is Change Log TimeLastModified (Section 2.2.1.8).
@ItemName: This value is a Change Log ItemName (Section 2.2.1.9)
@Int1: This value is a Change Log Int1 (Section 2.2.1.10)

Return Code Values: An integer which MUST be 0.
Result Sets: MUST NOT return any Result Sets.

3.1.4.51 proc_PatchLinkForFile

The proc_PatchLinkForFile Stored Procedure is called to update the directory and leaf name of a requested link for a file, to prepare for delayed fixup. It calls proc_DirtyDependents which marks all dependent documents and links as dirty. (see documentation for proc_DirtyDependents). The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_PatchLinkForFile(
    @SiteId              uniqueidentifier,
    @WebId               uniqueidentifier,
    @DirName             nvarchar(256),
    @LeafName            nvarchar(256),
    @OldLinkDirName      nvarchar(256),
    @OldLinkLeafName     nvarchar(256),
    @OldServerRel        bit,

    @ItemId                      int,
    @DocId                       uniqueidentifier,
    @Guid0                       uniqueidentifier,
    @Int0                        int,
    @FullUrl                     nvarchar(260),
    @EventType                   int,
    @ObjectType                  int,
    @TimeLastModifiedIncoming    datetime,
    @ItemName                    nvarchar(255) = NULL,
    @Int1                        int = NULL
);
```
@SiteId: The Site Collection Identifier of the Site Collection which contains the File.

@WebId: The Site Identifier of the Site.

@DirName: The directory containing the file. MUST NOT be NULL.

@LeafName: The file name. MUST NOT be NULL.

@OldLinkDirName: The directory containing the linked file, prior to running the stored procedure. MUST NOT be NULL.

@OldLinkLeafName: The linked file's name, prior to running the stored procedure. MUST NOT be NULL.

@OldServerRel: Indicates if old link is relative. Setting this to 'True' is meaningless unless @NewServerRel and @PatchPrefix are also 'True'.

@NewLinkDirName: Target directory for the linked file. MUST NOT be NULL.

@NewLinkLeafName: Target name for the linked file. MUST NOT be NULL.

@NewServerRel: Indicates if new link is relative. Setting this to 'True' is meaningless unless @OldServerRel and @PatchPrefix are also 'True'.

@PatchPrefix: If 'True', the whole URL of the directory is updated. Setting this to 'True' is meaningless unless @OldServerRel and @NewServerRel are also 'True'.

@DocUpdateFlags: Only of interest if the VPUTDOC_MIGRATIONSEMANTICS flag (8192) is set. This indicates that the client is performing a migration. If so, update the file's modification date/time information.

Return Code Values: An integer which MUST be listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful execution.</td>
</tr>
<tr>
<td>3</td>
<td>Cannot find the requested Document in the Site Collection, or the Site Collection does not exist.</td>
</tr>
<tr>
<td>1150</td>
<td>Error updating Link data.</td>
</tr>
</tbody>
</table>

Result Sets: MUST NOT return any Result Sets.

3.1.4.52 proc_PatchLinkForWeb

The proc_PatchLinkForWeb Stored Procedure is called to patch links within a Site, preparing it for delayed link Fixup. The directory name and leaf name parameters represent the store-relative parts of the site-relative URL's. The T-SQL syntax for the Stored Procedure is as follows:
PROCEDURE proc_PatchLinkForWeb(
    @SiteId uniqueidentifier,
    @WebId uniqueidentifier,
    @OldLinkDirName nvarchar(256),
    @OldLinkLeafName nvarchar(128),
    @NewLinkDirName nvarchar(256),
    @NewLinkLeafName nvarchar(128),
    @PatchFlags int = 0
);

@SiteId: The Site Collection Identifier of the Site Collection which contains the specified Site.

@WebId: The Site Identifier of the Site.

@OldLinkDirName: The directory name containing the linked Site. MUST NOT be NULL.

@OldLinkLeafName: The leaf name containing the linked Site. MUST NOT be NULL.

@NewLinkDirName: The new directory name for the linked Site. MUST NOT be NULL.

@NewLinkLeafName: The new leaf name for the linked Site. MUST NOT be NULL.

@PatchFlags: A 32-bit mask containing control flags. This MAY have zero, one, or two flags set. This parameter is optional with a default value of 0. The valid flags are:

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00000000</td>
<td>Default value, indicating that neither of the following situations is true.</td>
</tr>
<tr>
<td>0x00000001</td>
<td>Indicates directory names are more than one level above the leaf. The new Link directory name replaces part of the old Link directory name, and lower-level sites are updated. For example, the Domain name MAY change but the structure following that remains the same.</td>
</tr>
<tr>
<td>0x00000002</td>
<td>Indicates that the client is performing a migration, and the Site's modification date is updated.</td>
</tr>
</tbody>
</table>

Return Code Values: An integer which MUST be listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful execution.</td>
</tr>
<tr>
<td>3</td>
<td>Cannot find the requested Site in the Site Collection, or the Site Collection does not exist.</td>
</tr>
<tr>
<td>1150</td>
<td>Error updating Link data.</td>
</tr>
</tbody>
</table>

Result Sets: MUST NOT return any Result Sets.

3.1.453 proc_RefreshCheckout

The proc_RefreshCheckout Stored Procedure is called to renew the Short-Term Check Out on the specified Document for the specified User. The T-SQL syntax for the Stored Procedure is as follows:

PROCEDURE proc_RefreshCheckout(
    @SiteId uniqueidentifier,
    @DirName nvarchar(256),
);
@SiteId: The Site Collection Identifier of the Site Collection which contains the specified Checked Out Document.

@DirName: The directory name of the Document.

@LeafName: The leaf name of the Document.

@SystemID: The SystemID of the User who has Checked Out the Document.

@CheckoutTimeout: New timeout in minutes for Short-Term Check Out of the Document. It MUST NOT be NULL.

Return Code Values: An integer which MUST be listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Success. The requested operation was executed successfully.</td>
</tr>
<tr>
<td>2</td>
<td>File not found. A Checked Out Document corresponding to the specified @SiteId, @DirName, @LeafName, and @SystemID parameters could not be found.</td>
</tr>
<tr>
<td>5</td>
<td>An active User corresponding to the specified @SiteId and @SystemID parameters could not be found.</td>
</tr>
</tbody>
</table>

Result Sets: MUST return the following Result Sets under the specified conditions:

3.1.4.53.1 Document Metadata Result Set

This result set MUST be returned only if the Document specified by @SiteId, @DirName and @LeafName exists for the User specified by @SystemID. See [MS-WSSFO], section 2.2.5.6. All values in the column named {CacheParseId} MUST be NULL.

3.1.4.53.2 NULL Result Set

This result set MUST be returned only if the Document specified by @SiteId, @DirName and @LeafName exists for the User specified by @SystemID. See the NULL Result Set definition in [MS-WSSFO], section 3.1.4.4.3.

3.1.4.54 proc_RemoveJunctions

The proc_RemoveJunctions Stored Procedure is called to remove a value from the set of values of a Multi-valued lookup field of a specified List Item in a List. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_RemoveJunctions(
    @SiteId                 uniqueidentifier,
    @DirName                nvarchar(256),
    @LeafName               nvarchar(128),
    @FieldId                uniqueidentifier,
    @DeleteTransactionId    varbinary(16) = 0x,
);```

[MS-WSSDLIM] — v20120630
Windows SharePoint Services: Content Database Document and List Item Management Communications Protocol Specification

Copyright © 2012 Microsoft Corporation.

Release: July 16, 2012
@Level tinyint = 1,
@IsCurrentVersion bit = 1,
@CalculatedVersion int = 0
)

@SiteId: The Site Collection Identifier of the Site Collection which contains the specified List which contains the specified multivalued lookup field.

@DirName: The Directory Name of the specified List Item.

@LeafName: The Leaf Name of the specified List Item.

@FieldId: The Field Identifier of the specified multivalued lookup field.

@DeleteTransactionId: A parameter which identifies the Delete Transaction IDENTIFIER which encapsulated the actual delete operation. This is used so that multiple or hierarchical operations MAY be performed by the caller. This parameter SHOULD be set to its default value of 0x to appear in the UserDataJunctions view.

@Level: A Publishing Level value specifying the publishing status of this List Item.

@IsCurrentVersion: A bit flag specifying whether the specified row belongs to the Current Version of the List Item.

@CalculatedVersion: This Field keeps all live/nonhistorical rows for a List Item together. This value SHOULD be set to 0, as the UserDataJunctions view only contains non-historical List Items.

Return Values: proc_RemoveJunctions returns an integer Return Code which MUST be 0.

Result Sets: proc_RemoveJunctions MUST NOT return a Result Set.

3.1.4.55 proc_RenameHostHeaderSite

The proc_RenameHostHeaderSite Stored Procedure is called to change the URL of a host-named Site Collection to a new URL. It is executed in the configuration database and the content database. The HostHeader field in the Sites table is updated, and a row is added to the EventCache table. The procedure assumes that no Site already exists at that path.

The T-SQL syntax for the Stored Procedure is as follows:

```
PROCEDURE proc_RenameHostHeaderSite(
    @SiteId          uniqueidentifier,
    @HostHeader      nvarchar(128)
);
```

@SiteId: The Site Collection Identifier of the Site Collection. MUST NOT be NULL.

@HostHeader: The Internet Information Services (IIS) host header for the Internet Information Services (IIS) Web site of the Web application containing the Site Collection. The Host header is an optional property of an Internet Information Services (IIS) Web site. The caller of this procedure MUST specify @HostHeader = NULL, or enter a host header string.

Return Code Values: An integer which the protocol client MUST ignore.

Result Sets: MUST NOT return any Result Sets.
3.1.4.56 proc_RenameSite

The proc_RenameSite Stored Procedure is called to move a typical (that is, non-hostheader) Site to a different URL. All permutations of moves in root and non-root folder structure are supported; that is, 'sites/one' to, or 'sites/one' to 'sites/another/one', and so forth. The Stored Procedure assumes that no Site already exists at that path. It does not update the Site URL entry in the configuration database. The T-SQL syntax for the Stored Procedure is as follows:

```
PROCEDURE proc_RenameSite(
    @SiteId uniqueidentifier,
    @OldUrl nvarchar(260),
    @NewUrl nvarchar(260),
    @SiteFullUrl nvarchar(260) OUTPUT
);
```

@SiteId: The Site Collection Identifier of the Site Collection.

@OldUrl: The URL of the Site before the Stored Procedure is called. It MAY be the URL of the top-level site or any sub-level.

@NewUrl: The URL of the Site after it has been renamed. The new URL MUST be in the same Web application.

@SiteFullUrl: The response URL.

Return Code Values: An integer that MUST be listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful execution.</td>
</tr>
<tr>
<td>206</td>
<td>Error condition. Raised when the new URL of any of the Site's objects would be longer than 260 characters.</td>
</tr>
<tr>
<td>1003</td>
<td>Error condition. A Site with the identifier @SiteId was not found, or the operation caused a database error.</td>
</tr>
<tr>
<td>1150</td>
<td>Error condition. Returned by the Stored Procedure proc_DirtyListData when trying to get a list of Documents for which to modify the paths. The only way to generate this error is to have mismatching collations between the AllDocs table and the table variable with specific collations created.</td>
</tr>
</tbody>
</table>

Result Sets: MUST NOT return any Result Sets.

3.1.4.57 proc_SetNextId

The proc_SetNextId Stored Procedure is called to update the Next Available Identifier of an existing List. The Next Available Identifier is an integer identifier for the next new List Item in the List. The T-SQL syntax for the Stored Procedure is as follows:

```
PROCEDURE proc_SetNextId(
    @WebId uniqueidentifier,
    @ListId uniqueidentifier,
    @NextAvailableId int
);
```
@WebId: The Site Identifier of the Site which contains the specified List.

@ListID: The List Identifier of the List to update.

@NextAvailableId: The integer identifier for next new List Item in the List. If the @NextAvailableId is greater than the current value of the Next Available Identifier for the requested List, the proc_SetNextId MUST update the Next Available Identifier of the requested List to be @NextAvailableId. Otherwise, the proc_SetNextId MUST not update the Next Available Identifier of the requested List.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST NOT return any Result Sets.

3.1.4.58 proc_StartUndirtyList

The proc_StartUndirtyList Stored Procedure is called at the beginning of the bulk operation. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_StartUndirtyList(
    @WebId             uniqueidentifier,
    @ListId            uniqueidentifier,
    @CacheParseId      uniqueidentifier
);
```

@WebId: The Site Identifier of the Site.

@ListID: The List Identifier of the List.

@CacheParseId: GUID representing new bulk operation. This is used to assure that no one else has updated or started to update the Document during the bulk operation. MUST NOT be NULL.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST return the result sets described in the following sections.

3.1.4.58.1 Cache Parse Identifier Result Set

This result set contains the effective Cache Parse Identifier for the new link fixup operation. The T-SQL syntax for the result set is as follows:

```sql
tp_CacheParseId uniqueidentifier NOT NULL
```

tp_CacheParseId: GUID representing new link fixup operation. This is used to assure that no one else has updated or started to update the document during the link fixup operation.

3.1.4.59 proc_TakeOfflineDocument

The proc_TakeOfflineDocument Stored Procedure is called to take the last published version or major version of the Document offline, so that it is visible only to users with permission to edit the Document and no longer visible to users browsing the Site. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_TakeOfflineDocument(
```
@SiteId: The Site Collection Identifier of the Site Collection which contains the specified Document.

@WebId: The Site Identifier of the Site which contains the specified Document.

@DirName: The directory name containing the Document.

@LeafName: The leaf name of the Document.

@CreateVersion: A bit flag specifying whether the Document Library containing the Document has version numbering enabled. If version numbering is enabled for the Document Library, this parameter MUST be "1". This parameter MUST NOT be NULL.

@EnableMinorVersions: A bit flag specifying whether the Document Library containing the Document has Minor Version numbering enabled. If Minor Version numbering is enabled for the Document Library containing the Document, this parameter MUST be set to "1"; otherwise this parameter MUST be set to "0". If the Document is not in a Document Library, this parameter MUST be set to "0". This parameter MUST NOT be NULL.

@Moderated: A bit flag specifying whether the Document Library containing the Document is a moderated object. If Document Library containing the Document is a moderated object, this parameter MUST be set to "1". In all other cases, it MUST be "0". This parameter MUST NOT be NULL.

@UserId: The identifier for the current user who is requesting this operation. This value MUST refer to an existing user identifier for the specified Site Collection.

@MaxMajorVersion: The maximum number of major versions that the Document Library will track for any Document. This is a setting of the Document Library containing the specified Document.

@MaxMajorMinorVersion: The maximum number of major versions that can have associated minor versions of a Document in the Document Library. This is a setting of the Document Library containing the specified Document.

Return Code Values: An integer which MUST be listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful execution.</td>
</tr>
<tr>
<td>3</td>
<td>File Not Found. A Document having Publishing Level of published corresponding to the specified SiteId, WebId, DirName and LeafName was not found; or @EnableMinorVersions is &quot;0&quot; and @Moderated is &quot;0&quot;.</td>
</tr>
</tbody>
</table>

Result Sets: MUST return the following Result Sets under the specified conditions:
3.1.4.59.1 Document Metadata Result Set

This result set MUST be returned if the execution is successful. See [MS-WSSFO], section 2.2.5.6. All values in the column named {CacheParseId} MUST be NULL.

3.1.4.59.2 Event Receivers Result Set

This result set MUST be returned if the current version of the Document specified is published and the @WebId parameter specifies the Site Collection. This result set MUST contain one row for each event receiver registered for this Document with an Event Host Type (see [MS-WSSFO], section 2.2.3.5, Event Host Type of 3 (List Item)). See proc_GetEventReceivers.Event Receivers Result Set in section 2.2.5.9 of [MS-WSSFO].

3.1.4.59.3 NULL Result Set

This result set MUST be returned if the current version of the requested Document is published and the @WebId parameter is NULL. See the NULL Result Set definition in [MS-WSSFO], section 3.1.4.4.3.

3.1.4.59.4 Link Info Single Doc Result Set

This result set MUST be returned if the execution is successful. See proc_GetLinkInfoSingleDoc.Link Info Single Doc Result Set in section 3.1.4.25.1 of [MS-WSSFO].

3.1.4.60 proc_UndirtyListItem

The proc_UndirtyListItem Stored Procedure is called to update MetaInfo of the Document and clear the Dirty flag. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_UndirtyListItem(
    @SiteId uniqueidentifier,
    @DirName nvarchar(256),
    @LeafName nvarchar(128),
    @Level tinyint,
    @CacheParseId uniqueidentifier,
    @MetaInfo image
);  
```

**@SiteId**: The Site Collection Identifier of the Site Collection which contains the specified Document.

**@DirName**: The directory name of the specified Document.

**@LeafName**: The leaf name of the requested Document.

**@Level**: Indicates the Level of the List Item

**@CacheParseId**: GUID representing new bulk operation. This is used to assure that no one else has updated or started to update the Document during the bulk operation. MUST NOT be NULL.

**@MetaInfo**: A metadict for the Document. [MS-FPSE], Section 2.2.2.5.4.

**Return Code Values**: An integer which MUST be listed in the following table:
<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful execution.</td>
</tr>
<tr>
<td>3</td>
<td>Specified List or Site is not found.</td>
</tr>
<tr>
<td>1150</td>
<td>Specified cache parse IDENTIFIER is not correct; another cache parse IDENTIFIER is already in use.</td>
</tr>
</tbody>
</table>

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

### 3.1.4.61 proc_UpdateDirtyDocument

The `proc_UpdateDirtyDocument` Stored Procedure is called to store changes to an item that is a Folder or Document. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_UpdateDirtyDocument(
    @DocSiteId          uniqueidentifier,
    @DocDirName         nvarchar(256),
    @DocLeafName        nvarchar(128),
    @Level              tinyint,
    @DocContent         image,
    @DocSize            int,
    @CacheParseId       uniqueidentifier,
    @Dynamic            bit,
    @@DocWebId          uniqueidentifier OUTPUT,
    @@DocId             uniqueidentifier OUTPUT,
    @@DoclibRowId       int OUTPUT,
    @@DocDTW            datetime OUTPUT,
    @@DocVersion        int OUTPUT,
    @@DocUnghosted      bit OUTPUT,
    @ChunkSize          int,
    @@DocTextptr      varbinary(16) OUTPUT
);
```

- **@DocSiteId:** The Site Collection Identifier containing the Document to be updated. MUST NOT be NULL.
- **@DocDirName:** The directory name part of the URL for the item to be updated.
- **@DocLeafName:** The Leaf Name part of the URL for the item to be updated.
- **@Level:** The Publishing Level of the item to update. Valid values are in [MS-WSSFO], section 2.2.2.6.
- **@DocContent:** If the item to be updated has an associated Document stream, then @DocContent MUST contain the binary content to be stored.
- **@DocSize:** The size, in bytes, of the content in @DocContent.
- **@CacheParseId:** A GUID designating the cached Version of the item to update.
- **@Dynamic:** If the value is 1, this is a dynamic page; otherwise it is a static page.
- **@@DocWebId:** An output variable providing the Site IDENTIFIER containing the item updated.
- **@@DocId:** An output variable providing the Document IDENTIFIER of the item updated.
@@DoclibRowId: An output variable providing the row identifier of the item within the containing Document Library or List, if applicable.

@@DocDTW: An output variable providing a timestamp in Coordinated Universal Time (UTC) specifying when the last changes were made to the Document stream. If the @DocContent content was updated, @@DocDTW will be the date and time that the Stored Procedure is called.

@@DocVersion: An output variable providing an incremented version of the item.

@@DocUnghosted: An output variable which if set to 1, indicates the item to be updated was uncustomized and is now customized (2).

@ChunkSize: Specifies the size of the chunk of the Document to be updated. If less than @DocSize, specifies that @@DocTextPtr MUST be returned.

@@DocTextptr: An output parameter containing a pointer set to the storage location of @DocContent if @DocContent is not NULL and @DocSize is greater than @ChunkSize. If @DocContent is NULL or @DocSize is less than @ChunkSize, @DocTextptr is set to NULL.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST NOT return any Result Sets.

3.1.4.62 proc_UpdateItemInNameValuePair

The proc_UpdateItemInNameValuePair Stored Procedure is called to update the Indexed Fields and their values of the specified List Item. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_UpdateItemInNameValuePair(
@SiteId uniqueidentifier,
@WebId uniqueidentifier,
@ListId uniqueidentifier,
@ItemId int,
@Level tinyint = 1,
@FieldId1 uniqueidentifier = NULL,
@FieldValue1 sql_variant = NULL,
@FieldId2 uniqueidentifier = NULL,
@FieldValue2 sql_variant = NULL,
@FieldId3 uniqueidentifier = NULL,
@FieldValue3 sql_variant = NULL,
@FieldId4 uniqueidentifier = NULL,
@FieldValue4 sql_variant = NULL,
@FieldId5 uniqueidentifier = NULL,
@FieldValue5 sql_variant = NULL,
@FieldId6 uniqueidentifier = NULL,
@FieldValue6 sql_variant = NULL,
@FieldId7 uniqueidentifier = NULL,
@FieldValue7 sql_variant = NULL,
@FieldId8 uniqueidentifier = NULL,
@FieldValue8 sql_variant = NULL,
@FieldId9 uniqueidentifier = NULL,
@FieldValue9 sql_variant = NULL,
@FieldId10 uniqueidentifier = NULL,
@FieldValue10 sql_variant = NULL,
@InsertIfUpdateFails int = 0,
@SelectFromUserData bit = 0
);
```
@SiteId: The Site Collection Identifier of the Site Collection which contains the specified List Item.

@WebId: The Site Identifier of the Site which contains the specified List Item.

@ListID: The List Identifier of the List which contains the specified List Item.

@ItemId: The Identifier of the specified List Item in the List.

@Level: The Publishing Level. The default value is 1.

@FieldId#: The GUID of the Indexed Fields. There are ten FieldID parameters numbered from 1 to 10. The default values are NULL.

@FieldValue#: The value of the Indexed Fields. There are ten FieldValue parameters numbered from 1 to 10. The default values are NULL.

@InsertIfUpdateFails: If this parameter's value is 1, the indexed fields and values will be inserted in the NameValuePair Table (Section 2.2.5.2) table, if the specified List Item cannot be found in the name value pair table. If List Item is found, the indexed fields and values will be updated. If this parameter is 0 and the List Item is not found, the indexed fields and values will not be updated. The default value is 0.

@SelectFromUserData: An input parameter. If it is set to 1 and @FieldId# is not NULL and @FieldId# is one of the Field Identifier values in the following table, then associated @FieldValue# is replaced with the corresponding column of the row in the AllUserData table specified by @ListId, @ItemId and @Level. The default value is 0. This parameter MUST NOT be NULL.

<table>
<thead>
<tr>
<th>Field Value (Column in the AllUserData Table)</th>
<th>Field Id</th>
</tr>
</thead>
<tbody>
<tr>
<td>tp_Author</td>
<td>{1df5e554-ec7e-46a6-901d-d85a3881cb18}</td>
</tr>
<tr>
<td>tp_Editor</td>
<td>{d31655d1-d5b-4511-95a1-7a99e9b75bf2}</td>
</tr>
<tr>
<td>tp_Created</td>
<td>{8c06beca-0777-48f7-91c7-6da68bc07b69}</td>
</tr>
<tr>
<td>tp_Modified</td>
<td>{28cf6b45-f48-462a-b5cd-27b6f9d2bd5f}</td>
</tr>
<tr>
<td>tp_HasCopyDestinations</td>
<td>{26d0756c-986a-48a7-af35-bf18ab85f4a}</td>
</tr>
<tr>
<td>tp_UIVersion</td>
<td>{7841bf41-43d0-4434-9f50-a673baef7631}</td>
</tr>
<tr>
<td>tp_ContentTypeId</td>
<td>{03e45e84-1992-4d42-9116-26756012634}</td>
</tr>
<tr>
<td>tp_CheckoutUserId</td>
<td>{3881510a-4e4a-4ee8-b102-8ee8e2d0dd4b}</td>
</tr>
</tbody>
</table>

Return Code Values: An integer which MUST be listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful execution.</td>
</tr>
<tr>
<td>87</td>
<td>The input parameters are incorrect.</td>
</tr>
</tbody>
</table>

Result Sets: MUST NOT return any Result Sets.
3.1.4.63 proc_UpdateOrderNumber

The proc_UpdateOrderNumber Stored Procedure is called to update the value of Item Order Field in an existing valid List Item to make the List Item be displayed at the specified position when a set of List Items in the List are sorted by the Item Order Field in ascending order. A set of List Items could be all List Items in the List, or all List Items that are associated with a particular meeting instance in the List or all List Items that are under a Folder in the List. Item Order Field is a Field with name "Order" and GUID {ca4addac-796f-4b23-b093-d2a3f65c0774}. When a List has Item Order Field and the set of List Items are sorted by Item Order Field in ascending order, the value of Item Order Field determines the displaying position of the List Item. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_UpdateOrderNumber(
    @SiteId uniqueidentifier,
    @WebId uniqueidentifier,
    @ListId uniqueidentifier,
    @BaseType int,
    @ItemId int,
    @ItemOrder int,
    @fMultipleMtgDataList bit,
    @RootFolderUrl nvarchar(256),
    @fUpdateDTM bit = 0,
    @InstanceId int = -3
);
```

@SiteId: The Site Collection Identifier of the Site Collection which contains the specified List Item.

@WebId: The Site Identifier of the Site which contains the specified List Item.

@ListID: The List Identifier of the List.

@BaseType: The list base type of the List that contains the List Item to be updated. See [MS-WSSFO], section 2.2.3.9

@ItemId: The integer identifier of the existing valid List Item to be updated.

@ItemOrder: The displaying position of the List Item when the set of List Items are sorted by the Item Order Field in ascending order. The first displaying position is "1", the second displaying position is "2", and so on. If @ItemOrder is less than 1, it is same as the first displaying position. If the @ItemOrder is greater than the total number of the set of List Items, it is same as the last displaying position. The @ItemOrder MAY be NULL. When @ItemOrder is NULL, proc_UpdateOrderNumber MUST use @ItemId times 100 as the value for Item Order Field in the List Item. When @ItemOrder is not NULL, the proc_UpdateOrderNumber MUST determine appropriate value for Item Order Field so that the List Item showed at the position specified by @ItemOrder when the set of List Items are sorted by Item Order Field in ascending order.

@fMultipleMtgDataList: A bit flag specifying whether the Site specified by @WebId is a meeting workspace site and the List specified by @List contains data for multiple meeting instances of a recurrent meeting within the Site. The @fMultipleMtgDataList MUST NOT be NULL. When a meeting is a recurrent meeting, there are multiple meeting instances and each meeting instance has its own integer identifier. When @fMultipleMtgDataList is 1, the set of List Items are all List Items that are associated with the meeting instance specified by @InstanceId.

@RootFolderUrl: The Directory Name under which the set of List Items are stored. When @BaseType is 1, @RootFolderUrl MUST NOT be NULL and the set of List Items are all List Items stored under the Directory Name.
@fUpdateDTM: A bit flag specifying whether to update the List's last modified datetime. It MUST NOT be NULL. The default value is 0. When its value is 1, proc_UpdateOrderNumber MUST update the last modified date time of the List using the current Coordinated Universal Time (UTC) date time.

@InstanceId: The integer identifier of a meeting instance in a recurring meeting. When a meeting is a recurring meeting, there are multiple meeting instances. @InstanceId is the integer identifier of a meeting instance in the recurring meeting. It MUST NOT be NULL. The default value is -3. When its value is -3 and @fMultipleMtgDataList is 1, the proc_UpdateOrderNumber MUST use the integer identifier of the meeting instance associated with the List Item as the meeting instance identifier.

Return Code Values: An integer which MUST be listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful execution.</td>
</tr>
<tr>
<td>13</td>
<td>The proc_UpdateOrderNumber cannot determine a value for Item Order Field to ensure the List Item be displayed at the specified position.</td>
</tr>
</tbody>
</table>

Result Sets: MUST NOT return any Result Sets.

3.1.4.64 proc_UpdateVersionVirusInfo

The proc_UpdateVersionVirusInfo Stored Procedure is called to update the latest virus scanning related information for the specified version of the Document. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_UpdateVersionVirusInfo(  
    @DocSiteId       uniqueidentifier,  
    @DocId           uniqueidentifier,  
    @Version         int,  
    @VirusVendorID   int,  
    @VirusStatus     int,  
    @VirusInfo       nvarchar(255)  
) ;
```

@DocSiteId: The Site Collection Identifier of the Site Collection containing the Document version to be updated.

@DocId: The Document Identifier of the Document version.

@Version: The User Interface (UI) Version of the Document to be updated.

@VirusVendorID: Specifies the vendor identifier of the virus scanner which processed this Document version.

@VirusStatus: A value specifying the current virus check status of this Document. Valid values are specified in [MS-WSSFO], section 2.2.3.15.

@VirusInfo: Contains provider-specific message returned by the virus scanner when it last processed the Document version.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST NOT return any Result Sets.
3.1.4.65  proc_UpdateView

The proc_UpdateView Stored Procedure is called to save modifications to the specified View. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_UpdateView(
    @SiteId                   uniqueidentifier,
    @ListId                   uniqueidentifier,
    @ViewId                   uniqueidentifier,
    @UserId                   int,
    @IsPersonalView           bit,
    @View                     ntext,
    @DisplayName              nvarchar(255),
    @ContentTypeId            varbinary(512),
    @ViewFlags                int,
    @ViewMask                 int,
    @Level                    tinyint,
    @BypassCheck              bit = 0,
    @bUpdateAllPersonalViews  bit = 0
);
```

@SiteId: The Site Collection Identifier of the Site Collection which contains the specified View for the specified List.

@ListId: The List Identifier of the List for the specified View.

@ViewId: The view Identifier of the View for which modifications will be saved.

@UserId: The user identifier that originally created the specified personal view. This Stored Procedure parameter MUST be NULL for shared views.

@IsPersonalView: This Stored Procedure parameter MUST be one of the following values:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>The Stored Procedure updates the shared view specified by the @ListId Stored Procedure parameter and the @ViewId Stored Procedure parameter.</td>
</tr>
<tr>
<td>1</td>
<td>The Stored Procedure updates the personal view specified by the @ListId Stored Procedure parameter and the @ViewId Stored Procedure parameter.</td>
</tr>
</tbody>
</table>

@View: A query expressed in Collaborative Application Markup Language (CAML) used when processing this view. See [MS-WSSCAML] for more information about CAML. If this stored procedure parameter is NULL, then the existing CAML query is not modified.

@DisplayName: The name for the newly created View.

@ContentTypeId: The new content type Identifier for the specified View. If this Stored Procedure parameter is NULL, then the existing Content Type Identifier for the specified View is not modified.

@ViewFlags: The new View Flags for the specified View. If this Stored Procedure parameter is NULL, then the existing View Flags are not modified and @ViewMask MUST be ignored.

@ViewMask: The view flags that will be removed from the existing set of view flags when applying the @ViewFlags. If this Stored Procedure parameter is NULL, then the view flags will be set to NULL.

@Level: The Publishing Level for the View specified by the @ViewId Stored Procedure parameter.
**@BypassCheck:** If this Stored Procedure parameter is 0 then the following validation steps MUST occur. If this Stored Procedure parameter is 1, then the following validation steps MUST NOT occur.

- If the View specified by @ViewId has been Deleted then the View MUST not be modified.
- If @IsPersonalView is set to 0 and the Document Version of the View specified is not the most recent Version, then the View MUST NOT be modified.
- If @IsPersonalView is set to 0 and the List specified by @ListId requires List Items to be Checked Out prior to editing them and the Publishing Level for the View’s Document is not Checked Out, then the View MUST not be modified.

**@bUpdateAllPersonalViews:** If the parameter is set to 1, then the Stored Procedure will modify the personal view specified by @ViewId regardless of the @UserId.

**Return Code Values:** An integer which MUST be listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-2147467259</td>
<td>The modifications for the specified View were not saved.</td>
</tr>
<tr>
<td>0</td>
<td>Successful execution.</td>
</tr>
<tr>
<td>1</td>
<td>The modifications for the specified View were not saved.</td>
</tr>
<tr>
<td>3</td>
<td>The modifications for the specified View were not saved because one of the following:</td>
</tr>
<tr>
<td></td>
<td>- The @Level Stored Procedure parameter supplied was NULL.</td>
</tr>
<tr>
<td></td>
<td>- The @BypassCheck Stored Procedure parameter is 0 and the Document specified by the @ViewId Stored Procedure parameter has been Deleted.</td>
</tr>
<tr>
<td></td>
<td>- The View specified by the @ViewId Stored Procedure parameter with the specified @UserId and @Level and @IsPersonalView for the List specified by the @ListId Stored Procedure parameter does not exist.</td>
</tr>
<tr>
<td>33</td>
<td>The modifications for the specified View were not saved because all of the following are true:</td>
</tr>
<tr>
<td></td>
<td>- The @BypassCheck Stored Procedure parameter is 0.</td>
</tr>
<tr>
<td></td>
<td>- The @IsPersonalView Stored Procedure parameter is 0.</td>
</tr>
<tr>
<td></td>
<td>- The Document specified by the @ViewId Stored Procedure parameter is not the most recent Document Version.</td>
</tr>
<tr>
<td>158</td>
<td>The modifications for the specified View were not saved because all of the following are true:</td>
</tr>
<tr>
<td></td>
<td>- The @BypassCheck Stored Procedure parameter was 0.</td>
</tr>
<tr>
<td></td>
<td>- The @IsPersonalView Stored Procedure parameter was 0.</td>
</tr>
<tr>
<td></td>
<td>- The Document specified by the @ViewId Stored Procedure parameter is not Checked Out.</td>
</tr>
<tr>
<td></td>
<td>- The List specified by the @ListId Stored Procedure parameter mandates that</td>
</tr>
<tr>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td></td>
<td>Documents MUST be Checked Out prior to making Document modifications.</td>
</tr>
<tr>
<td>212</td>
<td>The specified View was not modified because the Site Collection has its WRITELOCK Site Collection Flag bit set.</td>
</tr>
<tr>
<td>1816</td>
<td>The specified View was not modified because saving the modifications for the View would have exceeded the Site Collection Quota.</td>
</tr>
</tbody>
</table>

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

### 3.1.4.66 proc_UpdateVirusInfo

The proc_UpdateVirusInfo Stored Procedure is called to update the latest virus scanning information for the current version of the Document. The T-SQL syntax for the Stored Procedure is as follows:

```
PROCEDURE proc_UpdateVirusInfo(
    @DocSiteId        uniqueidentifier,
    @DocDirName       nvarchar(256),
    @DocLeafName      nvarchar(128),
    @DocVersion       int,
    @VirusVendorID    int,
    @VirusStatus      int,
    @VirusInfo        nvarchar(255),
    @DocContent       image,
    @DocSize          int,
    @ChunkSize        int,
    @DocTextptr       varbinary(16) OUTPUT
);
```

**@DocSiteId:** The Site Collection Identifier of the Site Collection containing the Document.

**@DocDirName:** The Directory Name of the Document location.

**@DocLeafName:** The Leaf Name of the Document location.

**@DocVersion:** The internal version number of the Document.

**@VirusVendorID:** Specifies the vendor identifier of the virus scanner which processed this Document.

**@VirusStatus:** A value specifying the current virus check status of this Document. Valid values are specified in [MS-WSSFO] section 2.2.3.18.

**@VirusInfo:** Contains provider-specific message returned by the virus scanner when it last processed the Document.

**@DocContent:** This MUST either contain the content of the Document, or be NULL. If the @DocContent is not NULL, then this stored procedure will replace the document stream with the new @DocContent and increases the version number by 1. The Site Collection size is then updated with the new size of the Site Collection.

**@DocSize:** Size in bytes of the Document.

**@ChunkSize:** Specifies the size in bytes of the portion of the Document Stream in @DocContent.
@DocTextptr: An output parameter containing a pointer set to the storage location of @DocContent if execution is successful, @DocContent is not NULL, and @DocSize is greater than @ChunkSize otherwise the proc_UpdateVirusInfo MUST NOT set the value of @DocTextptr.

**Return Code Values:** An integer which MUST be listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful execution.</td>
</tr>
<tr>
<td>3</td>
<td>Document not found or @DocContent is not NULL and Document Stream not found.</td>
</tr>
</tbody>
</table>

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

### 3.1.4.67 proc_UpdateWebPartLinks

The proc_UpdateWebPartLinks Stored Procedure is called during link fixup to update Web Part properties. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_UpdateWebPartLinks(
    @SiteId uniqueidentifier,
    @WebPartID uniqueidentifier,
    @Level tinyint,
    @AllUsersProperties image,
    @Source ntext
);
```

- **@SiteId:** The Site Collection Identifier of the Site Collection which contains the specified Web Part.
- **@WebPartID:** The GUID of the Web Part for which to update Web Part properties. MUST NOT be NULL.
- **@Level:** Indicates the Publishing Level of the Web Part Page containing the Web Part.
- **@AllUsersProperties:** A serialized representation of zero or more customizable properties on the Web Part. If this value is NULL, then default values will be used for all of the customizable properties on the Web Part.
- **@Source:** Serialized representation of zero or more properties of a Web Part in an alternative format used by an HTML editor. The properties can be personalized.

**Return Code Values:** An integer that MUST be listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful execution.</td>
</tr>
<tr>
<td>2</td>
<td>The Web Part specified by @SiteId, @WebPartID and @Level cannot be found.</td>
</tr>
<tr>
<td>212</td>
<td>The specified Site Collection is locked.</td>
</tr>
<tr>
<td>1359</td>
<td>Internal Error.</td>
</tr>
<tr>
<td>1816</td>
<td>The quota for the specified Site Collection has been exceeded.</td>
</tr>
<tr>
<td>-2147467259</td>
<td>Unknown Error.</td>
</tr>
</tbody>
</table>
Result Sets: MUST NOT return any Result Sets.

3.1.4.68  proc_UserHasDataItems

The proc_UserHasDataItems Stored Procedure is called to determine whether a particular User has created any List Items in the List specified. The T-SQL syntax for the Stored Procedure is as follows:

```sql
PROCEDURE proc_UserHasDataItems(
    @ListID          uniqueidentifier,
    @UserID          int,
    @InstanceID      int
);
```

@ListID: The List Identifier of the List. This parameter MUST NOT be NULL.

@UserID: Specifies the identifier of the User.

@InstanceID: Specifies the identifier of the instance of the list. If the specified List has only one instance, then the @InstanceID MUST equal NULL.

Return Code Values: An integer which MUST be listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>The User has not yet created any List Items or the specified @ListID does not exist or the specified @UserID does not exist.</td>
</tr>
<tr>
<td>1</td>
<td>The User has at least one List Item created in the specified List.</td>
</tr>
</tbody>
</table>

Result Sets: MUST NOT return any Result Sets.

3.1.4.69  proc_InsertItemIntoNameValuePairCollated

The proc_InsertItemIntoNameValuePairCollated stored procedure is called to insert indexed fields and their values for the specified list item for a specific collation. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_InsertItemIntoNameValuePairCollated(
    @SiteId             uniqueidentifier,
    @WebId              uniqueidentifier,
    @ListId             uniqueidentifier,
    @ItemId             int,
    @Collation          smallint,
    @Level              tinyint = 1,
    @FieldId1           uniqueidentifier = NULL,
    @FieldValue1        nvarchar(255)    = NULL,
    @FieldId2           uniqueidentifier = NULL,
    @FieldValue2        nvarchar(255)    = NULL,
    @FieldId3           uniqueidentifier = NULL,
    @FieldValue3        nvarchar(255)    = NULL,
    @FieldId4           uniqueidentifier = NULL,
    @FieldValue4        nvarchar(255)    = NULL,
    @FieldId5           uniqueidentifier = NULL,
    @FieldValue5        nvarchar(255)    = NULL,
    @FieldId6           uniqueidentifier = NULL,
    @FieldValue6        nvarchar(255)    = NULL,
)
```
@SiteId: The site collection identifier of the site collection which contains the specified list item.

@WebId: The site identifier of the site which contains the specified list item.

@ListID: The list identifier of the list which contains the specified list item.

@Item Id: The identifier of the specified list item in the list.

@Collation: The collation identifier of the collation for the specified list item in the list.

@Level: The publishing level. The default value is 1.

@FieldId#: The field identifiers of the indexed fields. There are ten FieldId parameters numbered from 1 to 10. The default values are NULL.

@FieldValue#: The value of the indexed fields. There are ten FieldValue parameters numbered from 1 to 10. The default values are NULL.

Return Code Values: An integer which MUST be listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful execution.</td>
</tr>
<tr>
<td>87</td>
<td>The input parameters are invalid.</td>
</tr>
</tbody>
</table>

Result Sets: MUST NOT return any result sets.

3.1.4.70 proc_UpdateItemInNameValuePairCollated

The proc_UpdateItemInNameValuePairCollated stored procedure is called to update the indexed fields and their values of the specified list item for a specific collation. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_UpdateItemInNameValuePairCollated(
    @SiteId               uniqueidentifier,
    @WebId                uniqueidentifier,
    @ListId               uniqueidentifier,
    @ItemId               int,
    @Collation            smallint,
    @Level                tinyint = 1,
    @FieldId1             uniqueidentifier = NULL,
    @FieldValue1          nvarchar(255) = NULL,
    @FieldId2             uniqueidentifier = NULL,
    @FieldValue2          nvarchar(255) = NULL,
    @FieldId3             uniqueidentifier = NULL,
) ;
```
@FieldValue3 nvarchar(255) = NULL,
@FieldId4 uniqueidentifier = NULL,
@FieldValue4 nvarchar(255) = NULL,
@FieldId5 uniqueidentifier = NULL,
@FieldValue5 nvarchar(255) = NULL,
@FieldId6 uniqueidentifier = NULL,
@FieldValue6 nvarchar(255) = NULL,
@FieldId7 uniqueidentifier = NULL,
@FieldValue7 nvarchar(255) = NULL,
@FieldId8 uniqueidentifier = NULL,
@FieldValue8 nvarchar(255) = NULL,
@FieldId9 uniqueidentifier = NULL,
@FieldValue9 nvarchar(255) = NULL,
@FieldId10 uniqueidentifier = NULL,
@FieldValue10 nvarchar(255) = NULL,
@InsertIfUpdateFails int = 0

@SiteId: The site collection identifier of the site collection which contains the specified list item.

@WebId: The site identifier of the site which contains the specified list item.

@ListId: The list identifier of the list which contains the specified list item.

@ItemId: The item identifier of the specified list item in the list.

@Collation: The collation identifier of the collation for the specified list item in the list.

@Level: The publishing level. The default value is 1.

@FieldId#: The field identifier of the indexed fields. There are ten FieldId parameters numbered from 1 to 10. The default values are NULL.

@FieldValue#: The value of the indexed fields. There are ten FieldValue parameters numbered from 1 to 10. The default values are NULL.

@InsertIfUpdateFails: If this parameter's value is 1, the indexed fields and values will be inserted into the NameValuePair Table (section 2.2.5.2), if the specified list item cannot be found in the NameValuePair table. If list item is found, the indexed fields and values will be updated. If this parameter's value is not zero and the list item is not found, the indexed fields and values will not be updated. The default value is 0.

Return Code Values: An integer which MUST be listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful execution.</td>
</tr>
<tr>
<td>87</td>
<td>The input parameters are invalid.</td>
</tr>
</tbody>
</table>

Result Sets: MUST NOT return any result sets.

3.1.5 Timer Events

None.
3.1.6 Other Local Events

None.

3.2 WSSDLIM Client Details

The front-end Web server acts as a client when it calls the back-end database server requesting execution of stored procedures.

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.

The front-end Web server can maintain the following sets of data for this protocol within object structures. There is no requirement for the state within these structures to be a complete representation of all data maintained on the Back-End Database Server, but can be populated as various requests to the Back-End Database Server are fulfilled. Data maintained on the front-end Web server can be discarded after individual sequences of requests have finished as part of a response for a higher level event.

- Configuration
- Site Collections
- Sites
- Lists
- List Items
- Documents
- Users
- Groups

3.2.2 Timers

A connection timeout timer is set up on the front-end Web server to govern the total connection time for any requests to the Back-End Database Server. The amount of time is governed by a timeout value configured on the front-end Web server for all Back-End Database Server connections.

3.2.3 Initialization

The Front-End Web Server MUST validate the user making the request before calling the Stored Procedures. The Site Collection identifier and the **User Identifier** for the user making the request are looked up by the front-end Web server before calling additional Stored Procedures.
3.2.4 Message Processing Events and Sequencing Rules

The front-end Web server handles each Stored Procedure with the same processing method of calling the Stored Procedure and waiting for the Result Code and any Result Sets that will be returned.

The front-end Web server can execute dynamically generated SQL queries against the Stored Procedures, or the Tables and Views used within the database. However, unless otherwise specified, any data addition, removal, or modification MUST occur only by calling the listed Stored Procedures. SQL queries MUST NOT attempt to add, remove, or update data in any Table or View in the Content or Configuration databases, unless explicitly required:

- As part of link fixup of a list item, the protocol client updates the AllUserData Table define in [MS-WSSFO], section 2.2.7.3, to store the updated field data.

- When updating the values for calculated fields on a list item, the protocol client directly updates the sql_variant column of AllUserData where the calculation results are stored.

- When enabling an indexed field on a list, the protocol client directly updates the NameValuePair Table (or one of the collated NameValuePair tables for textual fields, based on the collation order of the site that contains the list), copying the indexed field data from the AllUserData table for all items in the list. When disabling an indexed field on a list, the protocol client deletes all rows from the table.

- As part of updating item order in an ordered list, the protocol client updates the tp_ItemOrder column of the AllUserData table to 1.79E+308 for all items being reordered before invoking proc_UpdateOrderNumber on each of those list items.

3.2.5 Timer Events

If the connection timeout event is triggered, the connection and the Stored Procedure call fails.

3.2.6 Other Local Events

No other local events impact the operation of this protocol.
4 Protocol Examples

This section provides specific example scenarios. These examples describe in detail the process of communication between the various servers components involved in the Windows SharePoint Services deployment. In conjunction with the detailed protocol documentation described in the reference documents, this information is intended to provide a comprehensive view of how Windows SharePoint Services front-end Web servers communicate with both EUC and Back-End Database Server systems.

4.1 Associate a Category with a Document

This scenario is initiated when a category is associated to a Document. This can be achieved by adding the Categories Field to a Document Library and assigning a set of comma-delimited Categories to the Field for a particular Document.

![Diagram of protocol example](image-url)

**Figure 7: Associate a category with a document**

For simplicity's sake, this example assumes that:

- The Categories being associated with the Document are contained in the Site which contains the Document.
- The Document which is being associated with a Category doesn't contain any Links.

The following actions happen:

1. proc_FetchDocumentForUpdate
2. Result sets [MS-WSSFO] 3.1.4.15.1 - 3.1.4.15.15
3. Dynamic SQL query
4. Result set
5.1 Begin new transaction
5.2 proc_UpdateDocument
5.3 proc_AddDocToCategory
5.4 proc_DirtyDependents
5.5 proc_UpdateListItem
5.6 Commit transaction
5.7 proc_GetLinkInfoSingleDoc

Link info single doc result set and output parameters and transaction result set
1. The front-end Web server fetches the MetaData information and Document content for the specified Document by calling the proc_FetchDocumentForUpdate Stored Procedure using TDS.


3. The front-end Web server fetches information about the Fields associated to the Document and other information by building a query in SQL Syntax, which is sent using TDS.

4. The Back-End Database Server returns 1 Result Set containing the requested information.

5. The front-end Web server builds a transactional dynamic query in SQL Syntax to add the Categories to the Site. This query is sent using TDS.
   1. The query begins a new transaction.
   2. The query attempts to update the MetaData information and contents for the specified Document using the proc_UpdateDocument Stored Procedure. The Stored Procedure also deletes all old Categories for the Document.
   3. The query then attempts to associate the Categories to the Document by calling the proc_AddDocToCategory Stored Procedure for each such category.
   4. The query then records that all items that are dependent on the Document being updated need to be subsequently updated by marking them as Dirty using the proc_DirtyDependents Stored Procedure.
   5. The query then attempts to update the List Item corresponding to the Document in the Document Library using the proc_UpdateListItem Stored Procedure.
   6. The query rolls back the transaction if the previous actions were not successful, or it commits the transaction if they were successful.
   7. If the transaction is committed, it gets the Link information for Links associated with the Document using the proc_GetLinkInfoSingleDoc Stored Procedure defined in section 3.1.4.25 of [MS-WSSFO].

6. The Back-End Database Server returns one or two Result Sets depending on whether the transaction committed. If the transaction committed, it returns the Link Info Single Doc Result Set. If it was rolled back, it isn't returned. Additionally, it returns a Result Set which contains the Return Code and output parameters of Stored Procedures called within the transaction.

4.2 Remove Association of a Category from a Document

This scenario is initiated when an action to remove association of a category from a Document occurs. This can be achieved by adding the Categories Field to a Document Library and removing the specified Category from the Categories Field for the Document.
Figure 8: Remove association of a category from a document

For simplicity’s sake, this example assumes that:

- The Categories being associated with the Document are contained in the Site which contains the Document.
- The Document which is being associated with a category doesn't contain any links.

The following actions happen:

1. The front-end Web server fetches the MetaData information and Document content for the specified Document by calling the proc_FetchDocumentForUpdate Stored Procedure using TDS.

2. The Back-End Database Server returns Result Sets as listed in [MS-WSSFO] 3.1.4.15.1-3.1.4.15.15.

3. The front-end Web server fetches information about the Fields associated to the Document and other information by building a query in SQL Syntax, which is sent using TDS.

4. The Back-End Database Server returns one Result Set containing the requested information.

5. The front-end Web server builds a transactional dynamic query in SQL Syntax to remove the Categories to the Site. This query is sent using TDS.

   1. The query begins a new Transaction.
2. The query attempts to update the MetaData information and contents for the specified Document using the proc_UpdateDocument Stored Procedure. This Stored Procedure also deletes all old Categories for the Document.

3. The query then records that all items that are dependent on the Document being updated need to be subsequently updated by marking them as Dirty using the proc_DirtyDependents Stored Procedure.

4. The query then attempts to update the List Item corresponding to the Document in the Document Library using the proc_UpdateListItem Stored Procedure.

5. The query rolls back the transaction if the previous actions were not successful, or it commits the transaction if they were successful.

6. The Back-End Database Server returns one or two Result Sets depending on whether the transaction committed. If the transaction committed, it returns the Link information for Links associated with the Document using the proc_GetLinkInfoSingleDoc Stored Procedure defined in section 3.1.4.25.1 of [MS-WSSFO].

4.3 Add a Category to a Site

This scenario is initiated when a category is added to a Site. This can be achieved by calling the set service metainfo RPC method. The category is added by modifying the metadata key "vti_categories."

![Diagram](Figure 9: Add a category to a site)

For simplicity's sake, this example assumes that:
• The category being added to the Site isn't already associated with the Site; that is, it's a new category.

The following actions happen:

1. The front-end Web server fetches the MetaData information for the specified Site using the proc_GetWebMetaInfo Stored Procedure.

2. The Back-End Database Server returns Result Sets as listed in [MS-WSSFO] 3.1.4.36.1–3.1.4.36.7. One of the Result Sets returned is the Site Categories Result Set described in [MS-WSSFO], section 3.1.4.38.1, which contains the set of Categories currently associated with the specified Site.

3. The front-end Web server builds a transactional dynamic query in SQL Syntax to add the category to the Site. This query is sent using TDS.
   1. The query begins a new Transaction.
   2. The query attempts to set the MetaData information for the specified Site using the proc_SetWebMetaInfo Stored Procedure.
   3. The query then attempts to ensure all Categories that were already associated to the Site (as returned in the proc_GetWebMetaInfo.Site Categories Result Set) are still associated to the Site. It does this by calling the proc_AddCategoryToWeb Stored Procedure for each such category.
   4. The query then attempts to add the new category by calling the proc_AddCategoryToWeb Stored Procedure.
   5. The query commits the transaction.


4.4 Remove a Category from a Site

This scenario is initiated when a category is removed from a Site. This can be achieved by calling the set service metainfo RPC method. The category is removed by modifying the metadata key "vti_categories."
Figure 10: Remove a category from a site

For simplicity’s sake, this example assumes that:

- The category being removed from the Site is associated with the Site.
- The category being removed is not associated with any Document in the Site.

The following actions happen:

1. The front-end Web server fetches the MetaData information for the specified Site using the proc_GetWebMetaInfo Stored Procedure.

2. The Back-End Database Server returns Result Sets as listed in [MS-WSSFO] 3.1.4.36.1–3.1.4.36.7. One of the Result Sets returned is the Site Categories Result Set described in [MS-WSSFO], section 3.1.4.38.1, which contains the set of Categories currently associated with the specified Site.

3. The front-end Web server fetches all Documents that are associated with the specified category by calling the proc_ListDocsInCategory Stored Procedure.

4. The Back-End Database Server returns the proc_ListDocsInCategory.DocsCategory Result Set containing the Directory Name and Leaf Name of Documents associated with the specified category. The assumption here is that this Result Set has no Rows. Otherwise, all such Documents would need to be disassociated with the Category before continuing.

5. The front-end Web server attempts to delete the specified category from the Site by calling the proc_DeleteCategory Stored Procedure using TDS.

7. The front-end Web server builds a transactional dynamic query in SQL Syntax to save the updated MetaData for the Site and ensure all Categories except the specified category are associated with the Site. This query is sent using TDS.

1. The query begins a new Transaction.

2. The query attempts to set the MetaData information for the specified Site using the proc_SetWebMetaInfo Stored Procedure.

3. The query then attempts to ensure all Categories that were already associated to the Site (as returned in the proc_GetWebMetaInfo.Site Categories Result Set) except for the deleted category are still associated to the Site. It does this by calling the proc_AddCategoryToWeb Stored Procedure for each such category.

4. The query commits the transaction.


4.5 Change Log

The stored procedures in this protocol example do not require special sequencing when they are called. The example illustrated in the following shows an application calls proc_GetChanges to get a list of events and their metadata.

User uploads a document to a document library. This effectively adds a list item to the document library, which causes an event to be appended to the change log.

The application calls proc_GetChanges with the following parameter, with the intention to retrieve any events that involves an item being added and occurs between Coordinated Universal Time (UTC) 2008/02/07 and UTC 2008/02/08 on a particular list.

- Site Collection Id: '61854258-1D17-410E-8363-ADC6C0B5C6D4'
- Site Id: '2FF0E4EC-B41B-412E-AEDF-C796BBF0D905'
- List Id: '27AC1BC8-BAF5-418A-8634-F31A9A8886D5'
- Start Time Stamp: '2008-02-07'
- Start Change Log Id: NULL
- End Time Stamp: '2008-02-08'
- End Change Log Id: NULL
- Event Object Type Flags: 1 (means List Item)
- Event Type Flags: 4096 (Add)

The EventInformation result set is returned that contains the time stamp and the Change Log Identifier of the earliest event, similar to the following:

- The time stamp of the first Event in the Change Log is Coordinated Universal Time (UTC) 2008-02-06 22:10:08.460
The Identifier of the first Event in the Change Log is 1

The EventDetails result set is returned, which contains 1 event and its metadata

- **EventTime:** 2008-02-07 19:06:48.943
- **Id:** 2159
- **SiteId:** 61854258-1D17-410E-8363-ADC6C0B5C6D4
- **WebId:** 2FF0E4EC-B41B-412E-AEDF-C796BBF0D905
- **ListId:** 27AC1BC8-BAF5-418A-8634-F31A9A8886D5
- **ItemId:** 1
- **DocId:** 3705DD61-8DB6-4C7B-AF2B-571E45721F8C
- **Guid0:** NULL
- **Int0:** NULL
- **ContentTypeId:** NULL
- **ItemFullUrl:** Shared Documents/myfile.doc
- **EventType:** 4097
- **ObjectType:** 1
- **TimeLastModified:** 2008-02-07 19:06:47.000
- **Int1:** NULL

This result set shows that during the time period requested by the application, one event in the Change Log matches the search criteria. It is a document named "myfile.doc", which was added to this Document Library around 2008-02-07 19:06:48.943.

### 4.6 Link Fixup

In the following example, a single item is dirty in a list with Id 43E3226F-55A6-41BC-A194-9DD74AF4A1D5 in a site with Id 2EF8C46F-D91F-46C1-8D9B-9A24A62AA268 whose root folder is Lists/Links.

The protocol client sends:

```plaintext
SET NOCOUNT ON; EXEC proc_StartUndirtyList '2EF8C46F-D91F-46C1-8D9B-9A24A62AA268','43E3226F-55A6-41BC-A194-9DD74AF4A1D5','B470CF0A-150F-41BC-9329-68BB7d427DA6'; SELECT TOP 1000 U.nvarchar3,U.tp_DirName,U.tp_LeafName,U.tp_Level FROM UserData AS U INNER JOIN Docs AS D ON U.tp_SiteId=D.SiteId AND U.tp_DirName=D.DirName AND U.tp_LeafName=D.LeafName AND U.tp_Level=D.Level AND D.ListDataDirty = 1 WHERE U.tp_SiteId='4AB8AC94-9F0A-44D3-B7DC-A81A323A680D' AND (U.tp_DirName=N'Lists/Links' OR U.tp_DirName LIKE N'Lists/Links/%') AND U.tp_RowOrdinal=0 ORDER BY U.tp_DirName Asc, U.tp_LeafName Asc, U.tp_Level
```

To which the protocol server returns the following rowset:

- **nvarchar3:** /Shared%20Documents/old%20test.txt
Indicating that a list item with a (DirName, LeafName, Level) key of (List/Links, 1_.000, 1) has invalid field data in nvarchar3 that is \Shared\Documents\old\test.txt. Because fewer than 1000 rows were returned, the protocol client determines that it need not continue to perform link fixup on the list.

The protocol client then calls:

```
EXEC proc_GetListDataLinks '4AB8AC94-9F0A-44D3-B7DC-A81A323A6BDF','2EF8C46F-D91F-46C1-8D9B-9A24A62AA268',N'Lists/Links',N'1_.000',1,N'Lists/Links',N'1_.000',1,1
```

Causing the protocol server to respond with an empty Web List For Normalization Result Set and the following List Data Link Information Result Set:

- DirName: Lists/Links
- LeafName: 1_.000
- Level: 1
- FieldId: C29E077D-F466-4D8E-8BBE-72B66C5F205C
- TargetDirName: Shared Documents
- TargetLeafName: test.txt
- Type: 74
- Security: 85
- Dynamic: 83
- ServerRel: 1
- Type: 0
- PointsToDir: 0

The first result set indicates that the site has no subsites. The second result set specifies that the new link data should be Shared Documents/text.txt

This allows the protocol client to compute the correct field data and undirty the list item via the following request:

```
exec sp_executesql N'DECLARE @@iRet int;SET @@iRet=0;BEGIN TRAN;UPDATE UserData SET nvarchar3=@P1 WHERE tp_SiteId=''4AB8AC94-9F0A-44D3-B7DC-A81A323A6BDF'' AND tp_DirName=N''Lists/Links'' AND tp_LeafName=N''1_.000'' AND tp_Level=1 AND tp_RowOrdinal=0;EXEC @@iRet = proc_UndirtyListItem ''4AB8AC94-9F0A-44D3-B7DC-A81A323A6BDF'',N'Lists/Links',N''1_.000'',1,''B470CF0A-150F-41BC-9329-68BB7D427DA6'',NULL; IF @@iRet <> 0 GOTO done;done:IF @@iRet <> 0 ROLLBACK TRAN;ELSE BEGIN COMMIT TRAN;EXEC proc_FinishUndirtyList ''2EF8C46F-D91F-46C1-8D9B-9A24A62AA268'',''43E3226F-55A6-41BC-A194-
```

---

**[MS-WSSDLIM] — v20120630**

*Windows SharePoint Services: Content Database Document and List Item Management Communications Protocol Specification*

Copyright © 2012 Microsoft Corporation.

Release: July 16, 2012
which generates no rowsets.

4.7 Themes

![Diagram of Themes](image)

Figure 11: Themes

This diagram illustrates the process of applying a new Theme to a Site. The following is a more in-depth explanation of this messaging sequence process:

1. The Client's Web Browser retrieves the Theme XML File and Theme XSD File from the front-end Web server.

2. The Client's Web Browser displays the list of Theme Names from the Theme XML File to the end user.

3. Using the Client's Web Browser, the end user selects a Theme Name to be applied to the Site.
4. The Client's Web Browser requests the front-end Web server to apply the new Theme to the Site using the Theme Name.

5. For each Theme File found at the Theme Installation Path on the front-end Web server, the front-end Web server calls the `proc_AddGhostDocument` Stored Procedure so that all Theme Metadata is uploaded to the Content Database on the Back-End Database Server.

6. The front-end Web server then calls the `proc_LoadTheme` Stored Procedure to retrieve the Theme Files Information Result Set and the Theme INF File Information Result Set.

7. The front-end Web server then calls the `proc_ListThemes` and the `proc_DeleteUrl` (defined in [MS-WSSFO]) Stored Procedures to remove the previous Theme from the Site. For each row returned the by Theme Information Result Set returned by the `proc_ListThemes` Stored Procedure, the `proc_DeleteUrl` Stored Procedure is called to remove the previous Theme from the Site. The `proc_DeleteUrl` Stored Procedure is NOT be called for the Theme just applied to the Site.

8. The front-end Web server then applies the Theme data returned by the Theme Files Information Result Set and the Theme INF File Information Result Set from the `proc_LoadTheme` Stored Procedure to the current Page requested by the Client's Web Browser.

9. The front-end Web server returns the current Page to the Client's Web Browser.

### 4.8 Lists.asmx CheckoutFile SOAP method

This example describes the requests made and responses returned when a Client sends a Simple Object Access Protocol (SOAP) request to check out a requested document.

This scenario is initiated by a call to the SOAP method command CheckoutFile in lists.asmx. See [MS-LISTSWS] for reference. For simplicity's sake, this example assumes that a requested Document is not checked out by any one and the user has enough permission to check it out.

1. The client calls `lists.asmx CheckoutFile`.

2. The front-end Web server in turn asks to get the Document to the Back-End Database Server. It does this by first calling the `proc_FetchDocForRead` Stored Procedure using TDS. [MS-WSSFO], section 3.1.4.14.


4. The front-end Web server then requests to check out the Document. It does this by calling the `proc_CheckoutDocument` Stored Procedure using TDS. See [MS-WSSFO], section 3.1.4.4.

5. The Back-End Database Server returns a successful Return Code status and two Result Sets:

6. Link Info Result Set, which returns a list of all Forward Links and Backward Links for the Documents contained in the requested Site.

7. Document Metadata Result Set, which returns the Metadata for the Documents contained in the requested Site.

8. A SOAP HTTP Response is returned to the User, indicating if the Document is successfully checked out or not.
4.9 Versions.asmx GetVersions SOAP method

This example describes the requests made and responses returned when a Client sends a SOAP request to get all the Document Versions of a requested Document.

This scenario is initiated by a call to the SOAP method command GetVersions in versions.asmx. See [MS-VERSS] for reference. For simplicity's sake, this example assumes that a requested Document exists and the user have enough permission to get all its document versions.

The Client calls versions.asmx’s GetVersions

The front-end Web server in turn ask to get the Document to the Back-End Database Server. It does this by first calling the proc_GetDocsMetaInfo Procedure using TDS. See [MS-WSSFO], section 3.1.4.24.

The Back-End Database Server returns Result Sets for the Document specified by the client

The front-end Web server then requests to get the Document Versions of this Document. It does this by calling the proc_ListDocumentVersions Stored Procedure using TDS. See [MS-WSSFO], section 3.1.4.40.

The Back-End Database Server returns a successful Return Code status and one Result Sets:

Document Versions Metadata Result Set, which returns the Metadata for the Document Versions for the specific Document.

A SOAP HTTP Response is returned to the User, list all the Document Versions for the specific Document

4.10 Add Fields to Trigger Allocation of Additional Rows

This scenario is initiated when a List or a Wide List already contains the maximum number of Fields for a type per Row in the Content Database, and a Field of that type is added to the List or Wide List.
Figure 12: Adding just enough fields to trigger allocation of additional rows per list item

The following actions happen:

1. The front-end Web server builds a transactional dynamic query in T-SQL Syntax to add the Field to the List or Wide List and add an additional row per List Item in the Content Database. This query is sent using TDS.
   1. The query begins a new Transaction.
   2. The query attempts to add the Field to the List or Wide List using the proc_UpdateListFields Stored Procedure defined in [MS-WSSCCSP], section 3.1.5.64.
   3. The query gets the Audit Mask Information for the List or Wide List to which the Field is being added using the proc_GetAuditMaskOutput Stored Procedure.
   4. The query attempts to allocate an additional row per List Item, specifying an incremented row ordinal count for the row using the proc_AddNewRowOrdToList Stored Procedure.
   5. The query records that the newly added Field is in use in the List or Wide List using the proc_MapContentTypeToList Stored Procedure.
   6. The query rolls back the transaction if the previous actions were not successful, or it commits the transaction if they were successful.

2. The Back-End Database Server returns one Result Set which contains the Return Code of the actions in the query and the output parameters from the proc_GetAuditMaskOutput Stored Procedure.

4.11 Allocate Rows While Inserting an Item into a Wide List

This can happen if a Field that previously caused an additional row to be allocated per List Item was deleted, and a List Item is added to the Wide List. This scenario is initiated if a row needs to
allocated for a List Item in the Content Database if a row of a specific row ordinal in a Wide List doesn't correspond to any of the Fields in the Wide List.

Figure 13: Allocating new rows for a list item while inserting a list

For simplicity's sake, this example assumes:

- The Wide List does not contain any Indexed Fields, lookup fields, and Calculated Fields.
- The Wide List does not have any alerts associated with it.
- The Wide List is not a list of meeting attendees.
- The List Item being added does not require any delayed link fixup.

The following actions happen:

1. The front-end Web server builds a transactional dynamic query in SQL Syntax to add a List Item and any Rows not corresponding to any Fields to a Wide List. This query is sent using TDS.
   1. The query begins a new Transaction.
   2. The query attempts to add a Row for each row ordinal that has at least one Field corresponding to the Row using the proc_AddListItem Stored Procedure.
   3. The query attempts to add a Row for each row ordinal that does not have even a single Field corresponding to it using the proc_AddNewRowOrdToListItem Stored Procedure.
   4. The query rolls back the transaction if the previous actions were not successful, or it commits the transaction if they were successful.

2. The Back-End Database Server returns one Result Set which contains the Return Code of the actions in the query and the identifier of the List Item being added.
5 Security

5.1 Security Considerations for Implementers

Security for this protocol is controlled by the permissions to the databases on the Back-End Database Server, which is negotiated as part of the Tabular Data Stream [MS-TDS] protocol.

The database access account used by the front-end Web server must have access to the appropriate Content Database on the Back-End Database Server. If the account does not have the correct permissions, access will be denied when attempting to set up the [MS-TDS] connection to the Content Database, or when calling the Stored Procedures.

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

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

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

<1> Section 2.2.1.2: Windows SharePoint Services 3.0 sometimes set this value to NULL even if the Security Principal is active.

<2> Section 2.2.1.5: Windows SharePoint Services 3.0 sets this to value which is NOT NULL when a document content stream is reset.

<3> Section 2.2.1.9: Windows SharePoint Services 3.0 sets this to value which is NOT NULL when a document content stream is reset.

<4> Section 2.2.1.9: Windows SharePoint Services 3.0 sets this to value which is NOT NULL when a document content stream is reset.

<5> Section 3.1.4.4: Windows SharePoint Services 3.0 implementation did not follow the recommended behavior. SPWeb.InsertAlertEvent() will pass in an absolute URL pointing to the Site instead of the item in the @ItemFullUrl parameter.

<6> Section 3.1.4.32.4: Windows SharePoint Services 3.0 implementation did not follow the recommended behavior. Instead, {ContentModifiedSince} is set to 1 if any of the following is true: The document is a Dynamic Page. @ValidationType is set to either 1 or 2. @ValidationType is 3 and the last modified time of the specified Document Version matches the @IfModifiedSince or the value of @ClientId does not match the Document Identifier in the Back-End Database Server. In all other cases, it MUST be set to 0.

<7> Section 3.1.4.37.2: Windows SharePoint Services 3.0 returns the top 1000 Events.
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 162
  server 47
Add a category to a site example 167
Add field to trigger allocation of additional rows example 175
Allocate rows while inserting an item into a wide list example 176
AllUserData table structure 45
Applicability 15
Associate a category with a document example 164
Attribute groups - overview 46
Attributes - overview 46

B

Binary structures - overview 44

C

Capability negotiation 15
Category operations
  server 47
Category operations overview 14
Change Log ContentTypeId simple type 28
Change Log DocId simple type 22
Change log example 170
Change Log Guid0 simple type 23
Change Log Int0 simple type 26
Change Log ItemFullUrl simple type 30
Change Log ItemId simple type 20
Change Log ItemName simple type 35
Change Log ListId simple type 17
Change log operations
  server 48
Change log operations overview 14
Change Log SiteId simple type 40
Change Log TimeLastModified simple type 32
Change Log WebId simple type 40
Change tracking 180
Check in and check out operations
  server 48
Check in and check out operations overview 14

Client
  abstract data model 162
  initialization 162
  local events 163
  message processing 163
  overview 162
  sequencing rules 163
  timer events 163
  timers 162
  WSSDLIM interface 162
Collated NameValuePair table structures 46
Common data types
  overview 17
  Complex types - overview 46

D

Data model - abstract
  client 162
  server 47
Data types
  Change Log ContentTypeId simple type 28
  Change Log DocId simple type 22
  Change Log Guid0 simple type 23
  Change Log Int0 simple type 26
  Change Log Int1 simple type 37
  Change Log ItemFullUrl simple type 30
  Change Log ItemId simple type 20
  Change Log ItemName simple type 35
  Change Log ListId simple type 17
  Change Log SiteId simple type 40
  Change Log TimeLastModified simple type 32
  Change Log WebId simple type 40
  common 17
  Page Navigation Dependency Examples simple type 41
Data types - simple
  Change Log ContentTypeId 28
  Change Log DocId 22
  Change Log Guid0 23
  Change Log Int0 26
  Change Log Int1 37
  Change Log ItemFullUrl 30
  Change Log ItemId 20
  Change Log ItemName 35
  Change Log ListId 17
  Change Log SiteId 40
  Change Log TimeLastModified 32
  Change Log WebId 40
  Page Navigation Dependency Examples 41
Delete flag structure 43
Document flag structure 43

E

Elements - overview 46
  Event object type flag structure 41
  Event type flag structure 42
Events
  local - client 163
  local - server 162
  timer - client 163
  timer - server 161
Examples
  add a category to a site 167
  add field to trigger allocation of additional rows 175
  allocate rows while inserting an item into a wide list 176
  associate a category with a document 164
  change log 170
  link fixup 171