specifications and network programming art, and assumes that the reader either is familiar with the aforementioned material or has immediate access to it.

## Revision Summary

<table>
<thead>
<tr>
<th>Date</th>
<th>Revision History</th>
<th>Revision Class</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>07/13/2009</td>
<td>0.1</td>
<td>Major</td>
<td>Initial Availability</td>
</tr>
<tr>
<td>08/28/2009</td>
<td>0.2</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>11/06/2009</td>
<td>0.3</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>02/19/2010</td>
<td>1.0</td>
<td>Major</td>
<td>Updated and revised the technical content</td>
</tr>
<tr>
<td>03/31/2010</td>
<td>1.01</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>04/30/2010</td>
<td>1.02</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>06/07/2010</td>
<td>1.03</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>06/29/2010</td>
<td>1.04</td>
<td>Editorial</td>
<td>Changed language and formatting in the technical content.</td>
</tr>
<tr>
<td>07/23/2010</td>
<td>1.04</td>
<td>No change</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>09/27/2010</td>
<td>1.04</td>
<td>No change</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>11/15/2010</td>
<td>1.04</td>
<td>No change</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>12/17/2010</td>
<td>1.04</td>
<td>No change</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>03/18/2011</td>
<td>1.04</td>
<td>No change</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>06/10/2011</td>
<td>1.04</td>
<td>No change</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>01/20/2012</td>
<td>2.0</td>
<td>Major</td>
<td>Significantly changed the technical content</td>
</tr>
<tr>
<td>04/11/2012</td>
<td>2.0</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>2.0</td>
<td>No change</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
</tbody>
</table>
# Table of Contents

## 1 Introduction
- 1.1 Glossary....................................................................................................................... 8
- 1.2 References .................................................................................................................... 8
  - 1.2.1 Normative References .......................................................................................... 10
  - 1.2.2 Informative References ....................................................................................... 11
- 1.3 Protocol Overview (Synopsis) ...................................................................................... 11
  - 1.3.1 Event Operations .................................................................................................. 11
  - 1.3.2 Web Part Operations ........................................................................................... 12
  - 1.3.3 Workflow Operations ......................................................................................... 12
  - 1.3.4 Work Item Operations ......................................................................................... 12
- 1.4 Relationship to Other Protocols .................................................................................. 12
- 1.5 Prerequisites/Preconditions ....................................................................................... 13
- 1.6 Applicability Statement ............................................................................................. 13
- 1.7 Versioning and Capability Negotiation ......................................................................... 13
- 1.8 Vendor-Extensible Fields ........................................................................................... 13
- 1.9 Standards Assignments ............................................................................................. 15

## 2 Messages
- 2.1 Transport .................................................................................................................... 16
- 2.2 Common Data Types .................................................................................................. 16
  - 2.2.1 Simple Data Types and Enumerations ................................................................ 16
  - 2.2.2 Simple Data Types ............................................................................................... 16
    - 2.2.2.1 Context Collection Identifier .......................................................................... 16
    - 2.2.2.2 Context Identifier ........................................................................................... 16
    - 2.2.2.3 Context Object Identifier .............................................................................. 16
    - 2.2.2.4 Context Type Identifier .................................................................................. 16
    - 2.2.2.5 Event Receiver Source Identifier ................................................................. 16
  - 2.2.6 List Item Version ................................................................................................... 16
  - 2.2.7 Workflow Template Identifier .............................................................................. 16
  - 2.2.3 Bit Fields and Flag Structures .............................................................................. 16
    - 2.2.3.1 Event Receiver Source Type .......................................................................... 16
    - 2.2.3.2 Workflow Association Configuration ........................................................... 17
    - 2.2.3.3 Workflow Internal State .................................................................................. 18
    - 2.2.3.4 Workflow Status1 .......................................................................................... 18
  - 2.2.4 Enumerations ........................................................................................................ 19
    - 2.2.4.1 Sandboxed Solution Status ............................................................................ 19
  - 2.2.5 Binary Structures .................................................................................................. 19
    - 2.2.6.1 List Item Workflows Result Set ..................................................................... 19
    - 2.2.6.2 Solution Resource Usage Result Set ............................................................... 21
    - 2.2.6.3 Solution Resource Usage Processing Result Set ....................................... 21
    - 2.2.6.4 Web Parts Result Set ....................................................................................... 22
    - 2.2.6.5 Workflow Associations Result Set ................................................................. 23
    - 2.2.6.6 Work Items Result Set .................................................................................... 25
  - 2.2.7 Tables and Views ................................................................................................... 26
  - 2.2.8 XML Structures .................................................................................................... 26
    - 2.2.8.1 Namespaces ..................................................................................................... 26
    - 2.2.8.2 Simple Types .................................................................................................. 26
    - 2.2.8.3 Complex Types ............................................................................................... 26
    - 2.2.8.4 Elements .......................................................................................................... 26
3 Protocol Details .................................................................................................................. 28
  3.1 Server Details .................................................................................................................. 28
    3.1.1 Abstract Data Model ................................................................................................. 28
      3.1.1.1 Web Parts ........................................................................................................ 28
        3.1.1.1.1 Customizable and Personalizable Properties ............................................... 28
        3.1.1.1.2 Adding and Modifying a Web Part for All Users (Customization) ............ 28
        3.1.1.1.3 Adding a Web Part for All Users then modifying it uniquely for a particular User (Personalization) ....................................................... 28
        3.1.1.1.4 Adding a Web Part just for a particular User (Personal Web Part) .......... 29
        3.1.1.1.5 Versioning Web Parts Pages ..................................................................... 29
        3.1.1.1.6 Changing a Web Part Type Identifier ......................................................... 29
        3.1.1.1.7 Web Part Caching ....................................................................................... 30
      3.1.1.2 Workflow ........................................................................................................... 30
        3.1.1.2.1 Workflow Concepts .................................................................................. 30
        3.1.1.2.2 Workflow Reusability .............................................................................. 30
      3.1.1.3 Work Items ........................................................................................................ 30
      3.1.1.4 Event Receivers ............................................................................................... 30
        3.1.1.4.1 Event Receiver Concepts ....................................................................... 30
        3.1.1.4.2 Registering Event Receivers ................................................................... 30
        3.1.1.4.3 Scopes of Event Receivers ..................................................................... 30
        3.1.1.4.4 Sequences of Event Receivers ................................................................. 31
      3.1.1.5 Quota Management ......................................................................................... 31
      3.1.1.6 Sandboxed Solution Resource Usage Monitoring ........................................... 31
    3.1.2 Timers ....................................................................................................................... 31
    3.1.3 Initialization ............................................................................................................ 31
    3.1.4 Message Processing Events and Sequencing Rules .................................................. 31
      3.1.4.1 proc_AddNonListViewWebPart .................................................................... 31
      3.1.4.2 proc_AddNonListViewFormWebPartForUrl .............................................. 33
      3.1.4.3 proc_AddSolution ......................................................................................... 35
      3.1.4.4 proc_AddWebPart ......................................................................................... 36
      3.1.4.5 proc_AddWorkflow ................................................................. 38
      3.1.4.6 proc_AddWorkflowAssociation ................................................................. 40
      3.1.4.7 proc_AddWorkItem ....................................................................................... 41
      3.1.4.8 proc_ApplyViewToListWebPart ................................................................... 43
      3.1.4.9 proc_AutoCleanupWorkflows ................................................................. 45
      3.1.4.10 proc_AutoDropWorkflows ................................................................. 45
      3.1.4.11 proc_CancelDeclarativeWorkflows ................................................... 47
      3.1.4.12 proc_CancelWorkflow .............................................................................. 47
      3.1.4.13 proc_CommitUpdatedZoneIds ................................................................. 48
      3.1.4.14 proc_CompleteInProgressWorkItems ................................................... 48
      3.1.4.15 proc_CopyDefaultViewWebParts .......................................................... 49
      3.1.4.16 proc_CountWorkflowAssociations ......................................................... 50
        3.1.4.16.1 Count Workflow Associations Result Set ..................................... 50
      3.1.4.17 proc_CountWorkflows ............................................................................. 50
        3.1.4.17.1 Count Workflows Result Set ............................................................. 51
      3.1.4.18 proc_CountWorkflowsBatch ................................................................... 51
        3.1.4.18.1 Workflows Batch Result Set ............................................................... 52
      3.1.4.19 proc_CreateListViewPart ........................................................................ 52
3.1.4.20 proc_DeleteDocEventReceiver ........................................... 54
3.1.4.21 proc_DeleteEventReceiver .................................................. 56
3.1.4.22 proc_DeleteEventReceiversBySourceId ................................. 57
3.1.4.23 proc_DeleteInProgressWorkItems ........................................ 58
3.1.4.24 proc_DeleteSmartPagePersonalization .................................. 59
3.1.4.25 proc_DeleteWebPart .......................................................... 60
3.1.4.26 proc_DeleteWebPartPersonalization ..................................... 61
3.1.4.27 proc_DeleteWebPartWhileSaving ........................................ 62
3.1.4.28 proc_DeleteZoneWebPartsWhileSaving .................................. 63
3.1.4.29 proc_DisableAssociationsForTemplate .................................. 63
3.1.4.30 proc_DropWorkflow .......................................................... 64
3.1.4.31 proc_DropWorkflowAssociation ......................................... 64
3.1.4.32 proc_DropWorkItem .......................................................... 65
3.1.4.33 proc_EnableDeclarativeWorkflowAssociations ........................ 65
3.1.4.34 proc_EnumerateDocEventReceivers ..................................... 66
3.1.4.35 proc_EnumerateWebPartsForWeb ........................................ 66
3.1.4.36 proc_EnumerateWebPartsForWeb ........................................ 67
3.1.4.37 proc_FailOverInProgressWorkItems ..................................... 69
3.1.4.38 proc_GetAllResourceUsageForSiteToday ................................. 70
3.1.4.39 proc_GetAllWebPartsOnPage ............................................. 70
3.1.4.39.1 proc_GetAllWebPartsOnPage ........................................... 70
3.1.4.39.2 proc_GetAllWebPartsOnPage ........................................... 71
3.1.4.39.3 proc_GetAllWebPartsOnPage ........................................... 71
3.1.4.39.4 List Metadata, Result Set ............................................. 72
3.1.4.39.5 List Security Information, Result Set ............................... 72
3.1.4.40 proc_GetAverageDailyResourceUsagePerSite ......................... 72
3.1.4.41 proc_GetContextCollectionEventReceivers ........................... 73
3.1.4.41.1 Event Receivers Result Set ........................................... 73
3.1.4.42 proc_GetContextObjectEventReceivers .................................. 73
3.1.4.42.1 Event Receivers Result Set ........................................... 74
3.1.4.43 proc_GetDocEventReceivers ............................................... 74
3.1.4.43.1 Event Receivers Result Set ........................................... 75
3.1.4.44 proc_GetListItemWorkflows ............................................. 75
3.1.4.44.1 List Item Workflows Result Set ...................................... 76
3.1.4.45 proc_GetListItemWorkflowWithInstanceDataAndLock .................. 76
3.1.4.46 proc_GetListWebParts ..................................................... 77
3.1.4.46.1 List Web Parts Result Set ............................................. 78
3.1.4.47 proc_GetNextWebPartOrder ............................................... 79
3.1.4.48 proc_GetRecycleBinItemEventReceivers ................................ 80
3.1.4.48.1 Recycle Bin Item Result Set .......................................... 80
3.1.4.48.2 List Event Receivers Result Set ...................................... 81
3.1.4.48.3 Site Event Receivers Result Set ...................................... 81
3.1.4.49 proc_GetRunnableWorkItems ............................................ 81
3.1.4.49.1 Work Items Result Set ................................................ 83
3.1.4.50 proc_GetRunningWorkBatchCount ...................................... 83
3.1.4.51 proc_GetSiteResourceUsage ............................................. 83
3.1.4.51.1 Site Collection Resource Usage Result Set ......................... 84
3.1.4.52 proc_GetSiteSolutionResourceUsage ................................. 84
3.1.4.52.1 Site Solution Resource Usage Result Set ........................................ 84
3.1.4.53 proc_GetSolutionInfo ................................................................. 84
3.1.4.53.1 Solution Hash Information Result Set ........................................ 85
3.1.4.54 proc_GetSolutionResourceQuota .................................................. 85
3.1.4.54.1 Solution Resource Quota Result Set .......................................... 86
3.1.4.55 proc_GetSolutionResourceUsage ................................................... 86
3.1.4.55.1 Solution Resource Usage Result Set .......................................... 86
3.1.4.56 proc_GetSolutionResourceUsageDailyOrdinal ................................. 86
3.1.4.56.1 Solution Resource Usage Daily Ordinal Result Set .................... 87
3.1.4.57 proc_GetSolutionsData ................................................................. 87
3.1.4.57.1 Solution Data Result Set ........................................................ 87
3.1.4.58 proc_GetWFTemplatesLastModifiedForWeb .................................... 88
3.1.4.59 proc_GetWorkflowAssociations .................................................... 88
3.1.4.59.1 Workflow Associations Result Set ............................................ 89
3.1.4.60 proc_GetWorkflowDataForItem ..................................................... 89
3.1.4.60.1 Workflow Associations Result Set ............................................ 90
3.1.4.60.2 List Item Workflows Result Set ................................................ 90
3.1.4.61 proc_GetWorkItems ..................................................................... 90
3.1.4.61.1 Single Work Item Result Set .................................................... 91
3.1.4.61.2 Multiple Work Items Result Set ................................................ 91
3.1.4.62 proc_InsertContextEventReceiver ............................................... 91
3.1.4.63 proc_InsertDocEventReceiver ...................................................... 94
3.1.4.64 proc_InsertEventReceiver .......................................................... 95
3.1.4.65 proc_LogSolutionResourceUsage20 .............................................. 97
3.1.4.66 proc_LogSolutionResourceUsageDaily20 ...................................... 100
3.1.4.67 proc_LogSolutionResourceUsageWindowed20 ............................... 102
3.1.4.68 proc_ProcessSolutionResourceUsageLogData .................................. 106
3.1.4.68.1 Solution Resource Usage Log Processing Result Set .................. 106
3.1.4.69 proc_ProcessSolutionResourceUsageWindowedData ....................... 106
3.1.4.69.1 Windowed Solution Resource Usage Processing Result Set .......... 107
3.1.4.70 proc_ProvisionWebPart ............................................................... 107
3.1.4.71 proc_RemoveSolution ................................................................. 108
3.1.4.72 proc_ResetSiteResourceUsageWarnings ........................................ 108
3.1.4.73 proc_RestoreWebPartForDoc ....................................................... 108
3.1.4.74 proc_RevertInProgressWorkItem .................................................. 109
3.1.4.75 proc_RevertInProgressWorkItems ................................................ 110
3.1.4.76 proc_SetEventReceiverToSynchronous ........................................ 111
3.1.4.77 proc_TruuncateResourceUsageDaily ............................................. 111
3.1.4.78 proc_TruuncateResourceUsageLog ................................................ 112
3.1.4.79 proc_TruuncateResourceUsageWindowed ...................................... 112
3.1.4.80 proc_UpdateDataViewWhileSaving .............................................. 112
3.1.4.81 proc_UpdateDocEventReceiver .................................................... 113
3.1.4.82 proc_UpdateEventReceiver ........................................................ 115
3.1.4.83 proc_UpdateListFormWhileSaving ............................................... 117
3.1.4.84 proc_UpdateListItemWorkflowInstanceData .................................. 118
3.1.4.85 proc_UpdateListItemWorkflowLock .......................................... 121
3.1.4.86 proc_UpdateListViewFormWebPartSource ................................. 122
3.1.4.87 proc_UpdateListViewToDataViewForSite ..................................... 122
3.1.4.88 proc_UpdateListViewToDataViewForWeb ..................................... 123
3.1.4.89 proc_UpdateSiteResourceUsage ................................................... 123
3.1.4.90 proc_UpdateSolution ................................................................. 124
3.1.4.91 proc_UpdateSolutionResourceUsage ............................................ 125
3.1.4.92 proc_UpdateViewWhileSaving ...................................................... 125
1 Introduction

This document specifies the Windows SharePoint Services: Content Database Programmability Extensions Communications Protocol. This protocol specifies the communication sequences used by a protocol client to perform data query and update operations on a protocol server in relation to Web Part, event receiver, workflow, and work item data.

Sections 1.8, 2, and 3 of this specification are normative and can contain the terms MAY, SHOULD, MUST, MUST NOT, and SHOULD NOT as defined in RFC 2119. Sections 1.5 and 1.9 are also normative but cannot contain those terms. All other sections and examples in this specification are informative.

1.1 Glossary

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

- anonymous user
- Coordinated Universal Time (UTC)
- GUID
- object
- XML

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

- after event receiver
- All Users
- assembly
- assembly name
- attachment
- author
- back-end database server
- base view identifier
- binary payload
- CAML
- configuration database
- content database
- content type
- context collection
- context object
- context type
- current user
- current version
- customizable
- daily solution resource usage log
- Data View Web Part
- datetime
- declarative workflow association
- default list view
- default view
- delete transaction identifier
- deleted
- directory name
- display name
- document
- document library
The following terms are specific to this document:

**before event receiver**: A code routine that processes a synchronous event whose handler runs completely before the action that raised the event is finalized.

**declarative workflow**: A workflow that is created with XAML (Extensible Application Markup Language) files and does not require precompiled code to run.

**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-WSSCADM] Microsoft Corporation, "Windows SharePoint Services Content Database Administrative Communications Protocol Specification".


1.2.2 Informative References


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


[MS-WSSO] Microsoft Corporation, "Windows SharePoint Services Overview".

1.3 Protocol Overview (Synopsis)

This protocol specifies the communication between the front-end Web server and the back-end database server. This communication satisfies requests associated with events, Web Parts, workflows, and work items. This client/server protocol uses the Tabular Data Stream Protocol as defined in [MS-TDS] as its transport between the front-end Web server, and the back-end database server.

1.3.1 Event Operations

The protocol specifies methods for creating, retrieving, manipulating and deleting events. When client requests for event information are sent to the front-end Web server, the front-end Web server...
sends a series of stored procedure calls to the back-end database server for the requested information. The stored procedures return data which in turn can be used for further calls to other stored procedures. The front-end Web server turns the values in the return codes and result sets into the data and metadata for the events requested by the client, and returns it to the client using the same protocol used by the initial request.

### 1.3.2 Web Part Operations

The protocol specifies methods for creating, retrieving, manipulating, and deleting Web Parts. When client requests for Web Part information are sent to the front-end Web server, the front-end Web server sends a series of stored procedure calls to the back end database server for the requested information. The stored procedures return data which in turn can be used for further calls to other stored procedures. The front-end Web server turns the values in the return codes and result sets into the data and metadata for the Web Parts requested by the client, and returns it to the client using the same protocol used by the initial request.

### 1.3.3 Workflow Operations

The protocol specifies methods for creating, retrieving, manipulating, and deleting workflows. When client requests for workflow information are sent to the front-end Web server, it responds with a series of stored procedure calls to the back end database server for the requested information. The stored procedures return data which in turn can be used for further calls to other stored procedures. The front-end Web server turns the values in the return codes and result sets into the data and metadata for the workflow requested by the client, and returns it to the client using the same protocol used by the initial request.

### 1.3.4 Work Item Operations

The protocol specifies methods for creating, retrieving, manipulating and deleting work items. When client requests for work item information are sent to the front-end Web server, it responds with a series of stored procedure calls to the back end database server for the requested information. The stored procedures return data which in turn can be used for further calls to other stored procedures. The front-end Web server turns the values in the return codes and result sets into the data and metadata for the work items requested by the client, and returns it to the client using the same protocol used by the initial request.

### 1.4 Relationship to Other Protocols

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

![Diagram showing the transport stack](https://via.placeholder.com/150)

**Figure 1: This protocol in relation to other protocols**
1.5 Prerequisites/Preconditions

The operations described by the protocol operate between a front-end Web server 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 front-end Web server has appropriate permissions to call the stored procedures 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

This protocol has the following vendor extensible fields:

- **sandboxed solution hash data** – A binary structure that contains a hash of a sandboxed solution or the sandboxed solution validators for the sandboxed solution. This binary can be passed into or retrieved by the following stored procedures and result sets:
  - proc_AddSolution
  - proc_RemoveSolution
  - proc_UpdateSolution
  - Solution Hash Information Result Set
  - Solution Data Result Set

- Web Part customizable and personalizable properties – A binary structure that contains zero or more serialized customizable or personalizable properties for a Web Part. This binary can be passed into or retrieved by the following stored procedures and result sets:
  - proc_AddNonListViewFormWebPartForUrl
  - proc_AddWebPart
  - proc_CreateListViewPart
  - proc_UpdateWebPart
  - proc_UpdateWebPartProps
  - proc_UpdateWebPartWhileSaving
  - List Web Parts Result Set
  - Web Parts Result Set
workflow instance data – A binary payload that contains the state of a workflow. This binary can be passed into or retrieved by the following stored procedures and result sets:

- proc_UpdateListItemWorkflowInstanceData
- List Item Workflows Result Set

work item binary payload – A binary payload stored with a work item that can be used by the protocol client that runs the work item. This binary can be passed into or retrieved by the following stored procedures and result sets:

- proc_AddWorkItem
- proc_UpdateWorkItem
- Work Items Result Set

work item text payload – A text payload stored with a work item that can be used by the protocol client that runs the work item. This string can be passed into or retrieved by the following stored procedures and result sets:

- proc_AddWorkItem
- proc_UpdateWorkItem
- Work Items Result Set

workflow modification data – XML that contains data about a workflow. See Workflow Modifications for schema information. This XML can be passed into or retrieved by the following stored procedures and result sets:

- proc_UpdateListItemWorkflowInstanceData
- List Item Workflows Result Set

workflow association data – XML that contains information about a workflow association. This XML can be passed into or retrieved by the following stored procedures and result sets:

- proc_AddWorkflowAssociation
- proc_UpdateWorkflowAssociation
- Workflow Associations Result Set

Web Part cache data – A binary payload that contains cached information about a web part. This data can be passed into or retrieved by the following stored procedures and result sets:

- proc_AddNonListViewFormWebPartForUrl
- proc_UpdateWebPartCache
- List Web Parts Result Set
- Web Parts Result Set

Vendors are free to choose their own values for these fields. This protocol specifies no mechanism for guaranteeing uniqueness of vendor-specific values for these fields.
1.9 Standards Assignments

None.
2  Messages

2.1  Transport

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

2.2  Common Data Types

This section contains common definitions used by this protocol.

2.2.1  Simple Data Types and Enumerations

2.2.2  Simple Data Types

2.2.2.1  Context Collection Identifier

A GUID used to identify a context collection.

2.2.2.2  Context Identifier

A GUID used to identify an object or a group of objects related to an event receiver.

2.2.2.3  Context Object Identifier

A GUID used to identify the context object for the event host for which an event receiver is registered.

2.2.2.4  Context Type Identifier

A GUID used to identify a context type.

2.2.2.5  Event Receiver Source Identifier

A GUID used to identify an event receiver source.

2.2.2.6  List Item Version

A 4-byte integer counter incremented any time a change is made to the properties of a list item.

2.2.2.7  Workflow Template Identifier

A GUID used to identify the workflow template.

2.2.3  Bit Fields and Flag Structures

2.2.3.1  Event Receiver Source Type

This is a 4-byte integer that specifies the event receiver source of an event receiver. All valid values for this type are specified in the following table.
<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No specific event receiver source.</td>
</tr>
<tr>
<td>1</td>
<td>The Event Receiver Source is a content type.</td>
</tr>
<tr>
<td>2</td>
<td>The Event Receiver Source is a feature.</td>
</tr>
</tbody>
</table>

### 2.2.3.2 Workflow Association Configuration

A 32-bit mask describing the configuration of the workflow association. The only valid values of the Workflow Association Configuration mask are bitwise combinations of the values listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Name</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00000001</td>
<td>WFA_AUTO_START_ADD</td>
<td>The protocol server MUST create and run a workflow whenever a new list item is created in the list with which the workflow association is associated.</td>
</tr>
<tr>
<td>0x00000002</td>
<td>WFA_AUTO_START_CHANGE</td>
<td>The protocol server MUST create and run a workflow whenever a list item is modified in the list with which the workflow association is associated.</td>
</tr>
<tr>
<td>0x00000008</td>
<td>WFA_ALLOW_MANUAL_START</td>
<td>Users are allowed to manually create and run workflows created from the workflow association.</td>
</tr>
<tr>
<td>0x00000010</td>
<td>WFA_HAS_STATUS_COLUMN</td>
<td>The workflow association has a workflow status field.</td>
</tr>
<tr>
<td>0x00000020</td>
<td>WFA_LOCK_ITEM</td>
<td>When a front-end Web server is processing a workflow created from the workflow association, it MUST lock the workflow.</td>
</tr>
<tr>
<td>0x00000040</td>
<td>WFA_DECLARATIVE</td>
<td>The workflow association is a declarative workflow association.</td>
</tr>
<tr>
<td>0x00000080</td>
<td>WFA_NO_NEWWORKFLOWS</td>
<td>The server MUST NOT create any new workflows from the workflow association.</td>
</tr>
<tr>
<td>0x00000200</td>
<td>WFA_MARKED_FOR_DELETE</td>
<td>The workflow association has been marked for deletion by proc_AutoCleanupWorkflows.</td>
</tr>
<tr>
<td>0x00001000</td>
<td>WFA_COMPRESS_INSTANCEDATA</td>
<td>The workflow instance data of workflows created from the workflow association is compressed.</td>
</tr>
<tr>
<td>0x00008000</td>
<td>WFA_ALLOW_ASYNCMANUALSTART</td>
<td>If a workflow created from the workflow association cannot be manually started synchronously because the number of running workflows exceeds the configured limit, it will be deferred for later processing. If this flag is not set, the workflow will always be started synchronously, regardless of the number of running workflows.</td>
</tr>
</tbody>
</table>
### 2.2.3.3 Workflow Internal State

A 32-bit mask describing the state of the workflow. The only valid values of the Workflow Internal State mask are bitwise combinations of the values listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Name</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00000001</td>
<td>WFS_LOCKED</td>
<td>A front-end Web server has locked the workflow for processing. No other front-end Web server can process the workflow.</td>
</tr>
<tr>
<td>0x00000002</td>
<td>WFS_RUNNING</td>
<td>A front-end Web server is processing the workflow.</td>
</tr>
<tr>
<td>0x00000004</td>
<td>WFS_COMPLETETED</td>
<td>The workflow has completely processed. No further processing can be done.</td>
</tr>
<tr>
<td>0x00000008</td>
<td>WFS_CANCELED</td>
<td>The workflow was canceled by a user. No further processing can be done.</td>
</tr>
<tr>
<td>0x00000040</td>
<td>WFS_FAULTING</td>
<td>The workflow has encountered an error and will be terminated.</td>
</tr>
<tr>
<td>0x00000080</td>
<td>WFS_TERMINATED</td>
<td>The workflow was terminated by an unrecoverable error before being completely processed. No further processing can be done.</td>
</tr>
<tr>
<td>0x00000100</td>
<td>WFS_SUSPENDED</td>
<td>The workflow was suspended. The workflow can resume running.</td>
</tr>
<tr>
<td>0x00000400</td>
<td>WFS_HASNEWEVENTS</td>
<td>The workflow has events that need to be processed.</td>
</tr>
<tr>
<td>0x00000800</td>
<td>WFS_NOTSTARTED</td>
<td>The workflow has not yet started running.</td>
</tr>
<tr>
<td>0x00001000</td>
<td>WFS_HASWAKEUPTIME</td>
<td>A work item has been created to resume processing the workflow.</td>
</tr>
</tbody>
</table>

### 2.2.3.4 Workflow Status1

An integer describing the status of the workflow. The following values are defined, but the field is vendor-extensible, and other values are allowed:

<table>
<thead>
<tr>
<th>Value</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>WFSTAT_NOTSTARTED</td>
<td>The workflow has not yet started running.</td>
</tr>
<tr>
<td>1</td>
<td>WFSTAT_FAILEDTOSTART</td>
<td>The workflow failed to start.</td>
</tr>
<tr>
<td>2</td>
<td>WFSTAT_INPROGRESS</td>
<td>A front-end Web server is processing the workflow.</td>
</tr>
<tr>
<td>3</td>
<td>WFSTAT_FAULTING</td>
<td>The workflow has encountered a faulting error.</td>
</tr>
<tr>
<td>4</td>
<td>WFSTAT_USERCANCEL</td>
<td>The workflow was canceled by a user.</td>
</tr>
<tr>
<td>5</td>
<td>WFSTAT_COMPLETETED</td>
<td>The workflow has completely processed.</td>
</tr>
<tr>
<td>6</td>
<td>WFSTAT_FAILEDTOSTART_RETRY</td>
<td>The workflow failed to start. Processing can be attempted again.</td>
</tr>
<tr>
<td>7</td>
<td>WFSTAT_FAULTING_RETRY</td>
<td>The workflow has encountered a faulting error. Processing can be attempted again.</td>
</tr>
</tbody>
</table>
2.2.4 Enumerations

2.2.4.1 Sandboxed Solution Status

The status of the sandboxed solution. The possible values are listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>The sandboxed solution has not been activated.</td>
</tr>
<tr>
<td>1</td>
<td>The sandboxed solution has been activated.</td>
</tr>
<tr>
<td>2</td>
<td>The sandboxed solution has been temporarily disabled because it exceeded its resource quota.</td>
</tr>
</tbody>
</table>

2.2.5 Binary Structures

None.

2.2.6 Result Sets

2.2.6.1 List Item Workflows Result Set

The List Item Workflows Result Set returns information about workflows created for List items. The T-SQL syntax for the result set is as follows:

```sql
Id                  uniqueidentifier,
TemplateId          uniqueidentifier,
ListId              uniqueidentifier,
SiteId              uniqueidentifier,
WebId               uniqueidentifier,
ItemId              int,
ItemGUID            uniqueidentifier,
TaskListId          uniqueidentifier,
AdminTaskListId     varbinary(16),
Author              int,
Modified            datetime,
Created             datetime,
StatusVersion       int,
Status1             int,
Status2             int,
Status3             int,
Status4             int,
Status5             int,
Status6             int,
Status7             int,
Status8             int,
Status9             int,
Status10            int,
TextStatus1         nvarchar(128),
TextStatus2         nvarchar(128),
TextStatus3         nvarchar(128),
TextStatus4         nvarchar(128),
TextStatus5         nvarchar(128),
Modifications       nvarchar(max),
InstanceData        varbinary(max),
InstanceDataSize    int,
```

[MS-WSSPROG2] — v20120630
Windows SharePoint Services: Content Database Programmability Extensions Communications Version 2 Protocol Specification

Copyright © 2012 Microsoft Corporation.

Release: July 16, 2012
InternalState int,
ProcessingId int;

**Id:** The workflow identifier of the workflow. This value MUST NOT be NULL.

**TemplateId:** The Workflow Template identifier of the workflow template from which the workflow was created. This value MUST NOT be NULL.

**ListId:** The List Identifier, as specified in [MS-WSSFO2], section 2.2.1.5, of the List containing the list item for which the workflow was created. This value MUST NOT be NULL.

**SiteId:** The site collection identifier of the site collection which contains the workflow. This value MUST NOT be NULL.

**WebId:** The Site Identifier, as specified in [MS-WSSFO2], section 2.2.1.11, of the site which contains the workflow.

**ItemGuid:** The item GUID of the list item.

**TaskListId:** The List Identifier, as specified in [MS-WSSFO2], section 2.2.1.5, of the workflow task list of the workflow.

**AdminTaskListId:** This column MUST be NULL.

**Author:** The user identifier of the user that created the workflow.

**Modified:** The date and time in UTC when the workflow was last modified. This value MUST NOT be NULL.

**Created:** The date and time in UTC when the workflow was created.

**StatusVersion:** The StatusVersion value for the workflow. This value MUST NOT be NULL.

**Status1:** The Workflow Status1 value for the workflow.

**Status2:** The protocol client MUST ignore this value.

**Status3:** The protocol client MUST ignore this value.

**Status4:** The protocol client MUST ignore this value.

**Status5:** The protocol client MUST ignore this value.

**Status6:** The protocol client MUST ignore this value.

**Status7:** The protocol client MUST ignore this value.

**Status8:** The protocol client MUST ignore this value.

**Status9:** The protocol client MUST ignore this value.

**Status10:** The protocol client MUST ignore this value.

**TextStatus1:** The protocol client MUST ignore this value.
2.2.6.2 Solution Resource Usage Result Set

The Solution Resource Usage Result Set contains resource usage values for a sandboxed solution. The T-SQL syntax for the result set is as follows:

```
SolutionId uniqueidentifier NOT NULL,
ResourceId uniqueidentifier NOT NULL,
RelativeDaysAgo int NOT NULL,
SampleCount int NOT NULL,
ResourceUsage float NOT NULL;
```

**SolutionId**: The identifier of the sandboxed solution.

**ResourceId**: The identifier of a monitored resource measure.

**RelativeDaysAgo**: The number of monitoring intervals since this resource usage value was collected.

**SampleCount**: The number of sample points taken for this resource usage value.

**ResourceUsage**: The resource usage value for the given sandboxed solution, monitored resource measure, and monitoring interval for resource usage.

2.2.6.3 Solution Resource Usage Processing Result Set

The Solution Resource Usage Processing Result Set is used when aggregating resource usage measurements. The T-SQL syntax for the result set is as follows:

```
SiteId uniqueidentifier NOT NULL,
SolutionId uniqueidentifier NOT NULL,
ResourceId uniqueidentifier NOT NULL,
StartTime datetime NOT NULL,
EndTime datetime NOT NULL,
SampleCount int NOT NULL,
ResourceUsage numeric NOT NULL,
Id bigint NOT NULL;
```

**SiteId**: The identifier of the site.
**SiteId:** The site collection identifier of the site collection that contains the sandboxed solution for this resource usage measurement.

**SolutionId:** The identifier of the sandboxed solution for this resource usage measurement.

**ResourceId:** The identifier monitored resource measure for this resource usage measurement.

**StartTime:** The beginning of the time interval, in the local time zone of the front-end Web server, corresponding to this resource usage measurement.

**EndTime:** The end of the time interval, in the local time zone of the front-end Web server, corresponding to this resource usage measurement.

**SampleCount:** The number of sample points taken for this resource usage measurement.

**ResourceUsage:** The resource usage measurement for the given sandboxed solution and monitored resource measure.

**Id:** The identifier for this resource usage measurement.

### 2.2.6.4 Web Parts Result Set

Web Parts Result Set returns properties of the Web Parts. There MUST be one row per Web Part in this Result Set. The T-SQL syntax for the result set is as follows:

```sql
tp_ID uniqueidentifier,
tp_ListId uniqueidentifier,
tp_Type tinyint,
tp_Flags int,
tp_DisplayName nvarchar(255),
tp_Version int,
{DocumentUrl} nvarchar(385),
tp_PartOrder int,
tp_ZoneID nvarchar(64),
tp_IsIncluded bit,
tp_FrameState tinyint,
tp_WebPartTypeId uniqueidentifier,
tp_Assembly nvarchar(255),
tp_Class nvarchar(255),
tp_SolutionId uniqueidentifier,
tp_AllUsersProperties varbinary(max),
tp_PerUserProperties varbinary(max),
tp_WebPartIdProperty nvarchar(255),
tp_Cache varbinary(max),
tp_Source nvarchar(max);
```

**tp_ID:** The Web Part Identifier, as specified in [MS-WSSFO2], section 2.2.1.15. This value MUST NOT be NULL.

**Tp_ListId:** The List Identifier, as specified in [MS-WSSFO2], section 2.2.1.5, of the List to which the Web Part refers.

**Tp_Type:** The Page Type, as specified in [MS-WSSFO2], section 2.2.3.14, of the Web Part Page that contains the Web Part.

**Tp_Flags:** The View Flags, as specified in [MS-WSSFO2], section 2.2.2.12, of the Web Part.
**Tp_DisplayName:** The display name of the Web Part.

**Tp_Version:** This value MUST be ignored.

**{DocumentUrl}:** The store-relative form URL of the Web Part. This value MUST NOT be NULL.

**Tp_PartOrder:** The Web Part zone index of the Web Part.

**Tp_ZoneID:** The Web Part zone identifier of the Web Part.

**Tp_IsIncluded:** 1 if the Web Part is included the Web Part Page; 0 if the Web Part is not included. This value MUST NOT be NULL.

**Tp_FrameState:** The Web Part chrome state of the Web Part. This value MUST NOT be NULL.

**Tp_WebPartTypeId:** The Web Part type identifier of the Web Part.

**Tp_Assembly:** The fully qualified name of the assembly that implements the web part.

**Tp_Class:** The name of the .NET class that implements the Web Part.

**Tp_SolutionId:** The identifier of the sandboxed solution or site solution that installed the Web Part.

**Tp_AllUsersProperties:** A binary payload containing 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.

**Tp_PerUserProperties:** A binary payload containing zero or more personalizable properties on the Web Part. If this value is NULL, then default values will be used for all of the personalizable properties on the Web Part.

**Tp_WebPartIdProperty:** The HTML (HyperText Markup Language) ID attribute of the Web Part. May be NULL. If not NULL, it MUST be unique per Web Part Page.

**Tp_Cache:** Private data cache of the Web Part.

**Tp_Source:** The Web Part properties of the Web Part in WPV2:WebPart format (as specified in [MS-WPPS], section 2.2.4.2), WPV3:WebPart format (as specified in [MS-WPPS] section 2.2.4.3) or WebParts format (as specified in [MS-WPPS] section 2.2.3.1). The protocol client can determine which format is used by comparing the value against the schemas for the formats. The value will be NULL if the properties are compressed and stored in Tp_AllUserProperties and Tp_PerUserProperties.

### 2.2.6.5 Workflow Associations Result Set

The Workflow Associations Result Set returns Workflow associations, one per row. The T-SQL syntax for the result set is as follows:

```sql
Id uniqueidentifier NOT NULL,
BaseId uniqueidentifier NOT NULL,
ParentId varbinary(16),
Name nvarchar(255),
Description nvarchar(1023),
StatusFieldName nvarchar(64),
SiteId uniqueidentifier NOT NULL,
WebId varbinary(16),
ListId varbinary(16),
ContentTypeId varbinary(512),
```

[MS-WSSPROG2] — v20120630

*Windows SharePoint Services: Content Database Programmability Extensions Communications Version 2 Protocol Specification*

*Copyright © 2012 Microsoft Corporation.*

*Release: July 16, 2012*
InstanceCount                int,
TaskListId                   varbinary(16),
HistoryListId               varbinary(16),
TaskListTitle                nvarchar(255),
HistoryListTitle             nvarchar(255),
Author                       int,
Created                      datetime,
Modified                     datetime,
PermissionsManual            bigint,
Version                      int,
AutoCleanupDays              int,
InstantiationParams          nvarchar(max),
Configuration                int;

Id: The workflow association identifier of the workflow association.

BaseId: The Workflow Template identifier of the workflow template on which the workflow association is based.

ParentId: The workflow association identifier of the parent workflow association of the workflow association of the row.

Name: The display name of the workflow association.

Description: The description of the workflow association.

StatusFieldName: The display name of the workflow status field of the workflow association.

SiteId: The site collection identifier of the site collection containing the workflow association.

WebId: The Site Identifier (as specified in [MS-WSSFO2], section 2.2.1.11) of the Site containing the workflow association.

ListId: The List identifier (as specified in [MS-WSSFO2], section 2.2.1.5) of the list with which the workflow association is associated.

ContentTypeId: The Content Type Identifier (as specified in [MS-WSSFO2], section 2.2.1.1) of the Content type with which the Workflow is associated.

InstanceCount: The current number of active workflows created from the workflow association.

TaskListId: The List Identifier (as specified in [MS-WSSFO2], section 2.2.1.5) of the workflow Task List of the workflow association.

HistoryListId: The List identifier (as specified in [MS-WSSFO2], section 2.2.1.5) of the workflow history list of the workflow association.

TaskListTitle: The display name of the workflow task list of the workflow association.

HistoryListTitle: The display name of the workflow history list of the workflow association.

Author: The user identifier of the author of the workflow association.

Created: The date and time in UTC when the workflow association was created.

Modified: The date and time in UTC when the workflow association was last modified.
PermissionsManual: The WSS Rights Mask (as specified in [MS-WSSFO2], section 2.2.14) required to manually start any workflows created from the workflow association.

Version: The version of the workflow association.

AutoCleanupDays: The number of days after which completed workflows created from the workflow association will be deleted by the back-end database server.

InstantiationParams: The workflow association data of the workflow.

Configuration: The Workflow association Configuration value for the workflow association.

2.2.6.6 Work Items Result Set

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

```
DeliveryDate        datetime,
Type                uniqueidentifier,
SubType             uniqueidentifier,
Id                  uniqueidentifier,
SiteId              uniqueidentifier,
ParentId            uniqueidentifier,
ItemGuid            uniqueidentifier,
BatchId             uniqueidentifier,
WebId               uniqueidentifier,
UserId              int,
Created             datetime,
BinaryPayload       varbinary(max),
TextPayload         nvarchar(max),
InternalState       int;
```

DeliveryDate: A UTC datetime representing when a work item is scheduled for execution. MUST NOT be NULL.

Type: The work item type identifier of the work item type. MUST NOT be NULL.

SubType: The work item subtype identifier of the work item subtype.

Id: The work item identifier.

SiteId: The site collection identifier of the site collection.

ParentId: The work item parent identifier of the work item.

ItemGuid: An item identifier for an list item associated with the work item. SHOULD <1> be 0 if there is no associated item. MUST NOT be NULL.

BatchId: The work item batch identifier of the work item batch. MUST be NULL if and only if the work item is a timer job.

ItemGuid: The item GUID.

WebId: The Site identifier (as specified in [MS-WSSFO2], section 2.2.1.10) of the Site.

UserID: The User identifier (as specified in [MS-WSSFO2], section 2.2.1.12) of the user associated with the work item. MUST NOT be NULL.
**Created**: The date and time in UTC specifying when the server created the work item.

**BinaryPayload**: The work item binary payload.

**TextPayload**: The work item text payload.

**InternalState**: An integer bit field specifying the internal state of the Work Item. All valid values are specified in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00000001</td>
<td>The work item is marked as in progress work item.</td>
</tr>
<tr>
<td>0x00000002</td>
<td>The work item is marked as completed work item.</td>
</tr>
<tr>
<td>0x00000004</td>
<td>The work item is marked for automatic deletion.</td>
</tr>
<tr>
<td>0x00000008</td>
<td>The work item is marked for exponential backoff.</td>
</tr>
<tr>
<td>0x00000010</td>
<td>The work item is marked for throttled fetch.</td>
</tr>
</tbody>
</table>

### 2.2.7 Tables and Views

None.

### 2.2.8 XML Structures

The syntax of the definitions in this section use XML Schema as specified in [XMLSCHEMA1] and [XMLSCHEMA2].

#### 2.2.8.1 Namespaces

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Namespace URI</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td><a href="http://www.w3.org/2001/XMLSchema">http://www.w3.org/2001/XMLSchema</a></td>
<td>[XMLSCHEMA1], [XMLSCHEMA2]</td>
</tr>
</tbody>
</table>

#### 2.2.8.2 Simple Types

None.

#### 2.2.8.3 Complex Types

None.

#### 2.2.8.4 Elements

The following table summarizes the set of common XML schema element definitions in this specification.

#### 2.2.8.4.1 Workflow Modifications

This is an XML structure that stores data about a workflow. The structure is used to store and correlate a set of vendor-supplied **GUIDs** and vendor-supplied **XML**.

```
<s:element name="Mods">
```
Mod.SubId: A string containing a GUID in which any alphabetic characters MUST be in upper case. This GUID is a vendor-extensible field.

Mod.Id: A string containing a GUID in which any alphabetic characters MUST be in upper case. The GUID identifies the parent Mod element.

Mod.TemplateId: A string containing a GUID in which any alphabetic characters MUST be in upper case. This GUID is shared among Mod elements in Mods that share some vendor-extensible characteristic.

Mod.Data: Any string or valid XML. This is a vendor-extensible field.

Example:

```xml
<Mods>
  <Mod>
    <SubId>F9168C5E-CEB2-4FAA-B6BF-329BF39FA1E4</SubId>
    <Id>936DA01F-9ABD-4D9D-80C7-02AF85C822A8</Id>
    <Data>Data string<abc></Data>
  </Mod>
</Mods>
```

2.2.8.5 Attributes
None.

2.2.8.6 Groups
None.

2.2.8.7 Attribute Groups
None.
3 Protocol Details

3.1 Server Details

3.1.1 Abstract Data Model

This section describes a conceptual model of possible data organization which 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 the behavior described in this document.

The back-end database server maintains the following sets of data for this protocol within both a configuration database and one or more content databases. Data within the appropriate databases is maintained until updated or removed.

3.1.1.1 Web Parts

3.1.1.1.1 Customizable and Personalizable Properties

A Web Part defines a number of properties that can be modified to change how the Web Part behaves or renders. The properties are split into two groups, customizable and personalizable. These two groups of properties are stored in the back-end database server for each Web Part, and both sets of properties are used to instantiate and render a Web Part on a front-end Web server. It is up to the Web Part implementer to determine whether a property is customizable or personalizable. A property is customizable if all users accessing the Web Part MUST get the same value for the property. A property is personalizable if users accessing the Web Part MUST be able to modify the property to a value specific to each user.

3.1.1.1.2 Adding and Modifying a Web Part for All Users (Customization)

When a Web Part is added to the shared view of a Web Parts page a new entry for the Web Part is added into the back-end database server containing all the personalizable and customizable properties of the Web Part. For each version of a Web Parts page there is only one copy of the personalizable and customizable properties stored in the back end database server for the shared view of a Web Part. As a result, when two different users browse to the shared view of the same Web Parts page the same set of personalizable and customizable properties for the Web Part are returned, resulting in the same Web Part being rendered for each user. Modifying this copy of properties used to render the shared view of a Web Part is called customization, and all users browsing to the shared view of the Web Parts page will see the same customized Web Part.

3.1.1.1.3 Adding a Web Part for All Users then modifying it uniquely for a particular User (Personalization)

When a Web Part is added to the shared view of a Web Parts page and a user then accesses the shared view or personal view of the Web Parts page, the personalizable and customizable properties returned for the Web Part will be the same so the Web Part will render the same in both the shared view and personal view.

If the user then modifies the Web Part from the personal view of the Web Parts page, then all of the personalizable properties currently stored in the back-end database server for the Web Part are copied into a separate entry in the back-end database server for the Web Part that is associated with the particular user who modified the Web Part.
This process is called personalization, and it means there are now two copies of the personalizable properties for the Web Part in the back-end database server, one copy that is used when any user accesses the Web Part in the shared view of the Web Parts page or they access the Web Part in the personal view of the Web Parts page but have not yet personalized the Web Part, and a second copy that is used when the user who personalized the Web Part accesses the Web Part in the personal view of the Web Parts page.

Every time a different user personalizes the Web Part an additional copy of the personalizable properties are stored for the Web Part in the back end database server for that particular user. When a user accesses the personal view of a Web Parts page, personalizable and customizable properties for the Web Part will be returned. If the Web Part has not been personalized by this user then these properties will be the same ones that are returned if the user browsed to the shared view of the Web Parts page. If the Web Part has been personalized by this user then the personalizable properties will be a unique copy that is stored in the back end database server just for this user, the customizable properties will be the same ones that are returned when accessing the shared view of the Web Parts page. There is only one copy of the customizable properties of a Web Part for a particular version of a Web Parts page, there is one copy of the personalizable properties of a Web Part for each user who has personalized that Web Part on the Web Parts page.

3.1.1.1.4 Adding a Web Part just for a particular User (Personal Web Part)

When a Web Part is added to the personal view of a Web Parts page a new entry for the Web Part is added into the back-end database server containing all the personalizable and customizable properties of the Web Part, and the entry is associated with the particular user who added the Web Part. This is called a personal Web Part and it will only be returned when the user who added the Web Part is accessing the Web Parts page in personal view. No one else will ever have access to this personal Web Part. If a personal Web Part is modified the one copy of the personalizable and customizable properties for the Web Part in the back end database server will be updated, and again only the user who added the personal Web Part will see the changes when they access the personal view of the Web Parts page.

3.1.1.1.5 Versioning Web Parts Pages

Versioning can be configured per list or per document library to store multiple versions of a Web Parts page. If minor version control is enabled on a Web Parts page, and modifications are made to a Web Part on that Web Parts page, then the back-end database server creates and stores a new version of the Web Parts page. The changes will be attributed to the user who made the changes. When a new version of a Web Parts page is created, an additional copy of all the personalizable and customizable properties used to render the shared view of the Web Parts on the Web Parts page is also created in the back end database server, this allows the Web Parts for different versions of the same Web Parts page to be independently modified. If a Web Part has been personalized by a user an additional copy of that users personalizable properties is NOT created, when a new version of the Web Parts page is created, this means if there are multiple versions of a Web Parts page there is only one copy of a particular users personalizations that gets used when that user is accessing the personal view of different versions of the Web Parts page.

For more information, refer to [MS-WSSO], section 2.6, Versioning.

3.1.1.1.6 Changing a Web Part Type Identifier

If this protocol is used to change the Web Part type identifier of an existing Web Part then the metadata stored in the back-end database server for that Web Part is no longer valid and is deleted.
3.1.1.1.7 Web Part Caching

Web Parts can choose to cache data to improve their performance or behavior on subsequent renderings. If this protocol is used to modify the properties of an existing Web Part that change potentially invalidates data that the Web Part has cached so if any such cached data exists for the Web Part it is deleted.

3.1.1.2 Workflow

3.1.1.2.1 Workflow Concepts

A workflow template defines a particular process of operations. The definition structures the order of operation, constraints, timing, and actual operations of this process. For example, a process which defines and manages how fields are changed on a document is a workflow template.

3.1.1.2.2 Workflow Reusability

A Workflow is based on a Workflow association that is applied to a specific list or Content type. Similarly a Workflow association is based on a workflow template, one of several processes stored on the server.

In line with this hierarchy, a workflow template creates one or many Workflow associations and a Workflow association creates one or many Workflows. This enables a particular process of operations to be reused in many different contexts.

3.1.1.3 Work Items

A Work item represents a unit of work that is scheduled for execution at the time indicated by its Work Item Delivery Date. Information about work items is kept in back end database server. The Work Item information specifies what type of work the work items will perform, when they MUST run, and what objects are related to them. These work items can be run by a protocol client that iterates through them and performs the appropriate code based on the work item type. Thus, a protocol client that creates the Work item works in tandem with a protocol client that retrieves and runs them in the way they were intended to be performed. Work item entries identified by work item identifiers.

3.1.1.4 Event Receivers

3.1.1.4.1 Event Receiver Concepts

Event receivers are custom code for extending functionalities by reacting to Events. Registration information about Event Receivers is kept in back end database server. The registration information determines what Event Receivers are processed for an Event.

3.1.1.4.2 Registering Event Receivers

The Event Host MUST register an Event Receiver to handle Events. If the event receiver is registered by feature or content type then the event receiver event receiver source property MUST point to this feature or content type, otherwise it MUST be NULL.

3.1.1.4.3 Scopes of Event Receivers

Event receivers can be registered on event hosts of different scopes. When an event is fired, it bubbles from the innermost event host outwards. For example, when a list item is updated, the
server fires an item updating event on the parent list containing the list Item first then on the Site containing the parent list.

3.1.1.4.4 Sequences of Event Receivers

When there are more than one event receiver registered on an event host, the processing order of the event receivers is the numerical order of the sequence numbers (1) of these event receivers. The event receiver with the smallest sequence number (1) is processed first.

3.1.1.5 Quota Management

Event, Web Part, Workflow, and Work Item operations typically use, or free, disk space in the back end database server. To manage this limited resource, quota management features can be enabled to track disk space usage, and block Event, Web Part, Workflow, and Work Item operations that use additional disk space if a Site Collection has exceeded its quota limits. See [MS-WSSCADM], section 1.3.2, for more information about quota management.

3.1.1.6 Sandboxed Solution Resource Usage Monitoring

As part of enforcing solution resource usage quotas, the protocol server implements a series of logs: the immediate solution resource usage log, the windowed solution resource usage log, and the daily solution resource usage log. The protocol client uses these logs to track the resource usage values for sandboxed solutions. To enable efficient management of these logs, the protocol server maintains an ordinal used for monitoring resource usage over a given interval that is incremented after every monitoring interval for resource usage.

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 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_AddNonListViewFormPersonalization

The proc_AddNonListViewFormPersonalization stored procedure is called to add a personalization to an existing Web Part which is not a list view Web Part. The T-SQL syntax for the stored procedure is as follows:

```sql
```
PROCEDURE proc_AddNonListViewFormPersonalization(
    @SiteId uniqueidentifier,
    @DocId uniqueidentifier,
    @WebPartId uniqueidentifier,
    @UserId int,
    @PartOrder int,
    @ZoneId nvarchar(64),
    @IsIncluded bit,
    @FrameState tinyint,
    @UserProperties varbinary(max),
    @RequestGuid uniqueidentifier = NULL OUTPUT
);

@SiteId: The Site Collection identifier of the Site Collection which contains the Web Part to be personalized. This MUST NOT be NULL.

@DocId: The Document Identifier (as specified in [MS-WSSFO2], section 2.2.1.2) of the Document which contains the Web Part to be personalized. This MUST NOT be NULL.

@WebPartId: The Web Part identifier (as specified in [MS-WSSFO2], section 2.2.1.15) of the Web Part to be personalized. This MUST NOT be NULL.

@UserId: The User Identifier (as specified in [MS-WSSFO2], section 2.2.1.13) of the user which personalizes the Web Part. This MUST NOT be NULL.

@PartOrder: The Web Part Zone Index of the added Web Part.

@ZoneId: The Web Part Zone identifier of the Web Part zone in which to put the Web Part.

@IsIncluded: The Web Part Is Closed state of the added Web Part. This value MUST NOT be NULL.

@FrameState: The Web Part chrome state of the added Web Part. This MUST NOT be NULL.

@UserProperties: The Web Part properties to assign to this Web Part for the user specified by @UserId.

@RequestGuid: The optional request identifier for the current request.

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 completion.</td>
</tr>
<tr>
<td>1</td>
<td>The operation failed to complete.</td>
</tr>
<tr>
<td>2</td>
<td>The requested Web Part does not exist.</td>
</tr>
<tr>
<td>212</td>
<td>The specified Site Collection is Locked.</td>
</tr>
<tr>
<td>1816</td>
<td>The Quota for the specified Site Collection has been exceeded.</td>
</tr>
</tbody>
</table>

Result Sets: MUST NOT return any result sets.
3.1.4.2 proc_AddNonListViewFormWebPartForUrl

The proc_AddNonListViewFormWebPartForUrl stored procedure is called to add a Web Part that is not a List View Web Part or List Form Web Part to a given page. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_AddNonListViewFormWebPartForUrl(
    @SiteId uniqueidentifier,
    @DocDirName nvarchar(256),
    @DocLeafName nvarchar(128),
    @WebPartId uniqueidentifier,
    @ListId uniqueidentifier,
    @Type tinyint,
    @Flags int,
    @DisplayName nvarchar(255),
    @ContentTypeId varbinary(512),
    @Version int,
    @PartOrder int,
    @ZoneId nvarchar(64),
    @IsIncluded bit,
    @FrameState tinyint,
    @WebPartTypeId uniqueidentifier,
    @Assembly nvarchar(255),
    @Class nvarchar(255),
    @SolutionId uniqueidentifier,
    @AllUserProperties varbinary(max),
    @PerUserProperties varbinary(max),
    @WebPartIdProperty nvarchar(255),
    @Cache varbinary(max),
    @Source nvarchar(max),
    @UserId int = NULL,
    @Level tinyint = 1,
    @BaseViewId tinyint = NULL,
    @bHasFGP bit = NULL,
    @bDeleteUsersOtherWebParts bit = 0,
    @bRetainObjectIdentity bit = 0,
    @View varbinary(max) = NULL,
    @RequestGuid uniqueidentifier = NULL OUTPUT
);```

@SiteId: The Site Collection identifier of the Site Collection which contains the Web Part Page to which to add the Web Part. This MUST NOT be NULL.

@DocDirName: The directory name of the Web Part Page to which to add the Web Part. This MUST NOT be NULL.

@DocLeafName: The leaf name of the Web Part Page to which to add the Web Part. This MUST NOT be NULL.

@WebPartId: The Web Part identifier (as specified in [MS-WSSFO2], section 2.2.1.15) of the Web Part being added. MUST NOT be NULL.

@ListId: The List Identifier (as specified in [MS-WSSFO2], section 2.2.1.5) of the list with which to associate the Web Part.

@Type: The Page type (as specified in [MS-WSSFO2], section 2.2.3.14) for the list view. If @Type has a value of default view the view MUST be made the Default View for the list.
@Flags: The set of View Flags (as specified in [MS-WSSFO2], section 2.2.12) to be applied to the added Web Part.

@DisplayName: The Display Name of the Web Part being added.

@ContentTypeId: The Content Type Identifier (as specified in [MS-WSSFO2], section 2.2.1.1) of the list Items in the list to be displayed in the Web Part.

@Version: The version number of the Web Part to add.

@PartOrder: The Web Part Zone Index of the Web Part to add.

@ZoneId: The Web Part Zone identifier of the Web Part Zone of the Web Part being added.

@IsIncluded: The Web Part Is Closed State of the added Web Part.

@FrameState: The Web Part Chrome State of the added Web Part.

@WebPartTypeId: The Web Part type identifier of the Web Part being added. MUST NOT be NULL.

@Assembly: The fully qualified name of the assembly that implements the web part.

@Class: The name of the .NET class that implements the web part.

@SolutionId: The identifier of the sandboxed solution that implements the web part.

@AllUsersProperties: A binary payload containing zero or more customizable properties on the Web Part.

@PerUserProperties: A binary payload containing zero or more personalizable properties on the Web Part.

@WebPartIdProperty: The HTML (HyperText Markup Language) ID attribute of the Web Part. May be NULL. If not NULL, it MUST be unique per Web Part Page.

@Cache: Private data cache of the Web Part.

@Source: The Web Part properties of the Web Part in either wpv2:WebPart format (as specified in [MS-WPPS], section 2.2.3.2) or HTML format.

@UserId: The User Identifier (as specified in [MS-WSSFO2], section 2.2.1.13) for the current user.

@Level: The publishing level of the Web Part Page for the current user.

@BaseViewId: The base view identifier for this Web Part.

@bHasFGP: This parameter MUST be ignored by protocol server.

@bDeleteUsersOtherWebParts: This parameter specifies whether all Web Parts on the page registered to the user MUST be deleted before this Web Part is added. If set to "1" then all Web Parts on the Page defined by the @SiteId, @Level, @DocDirName, and @DocLeafName parameters and registered to the user identified by @UserId MUST be deleted before this Web Part is added. If set to "0" then other Web Parts MUST NOT be modified.

@bRetainObjectIdentity: If this parameter is set to 1 and a Web Part identified by the @SiteId, @WebPartId and @Level parameters exists in a Web Part Page different from the Web Part Page...
identified by the @SiteId, @DocDirName and @DocLeafName parameters, the server MUST update the existing Web Part and move it to the Web Part Page rather than adding a new web part.

@View: The CAML XML for the View to be applied to the Web Part.

@RequestGuid: The optional request identifier for the current request.

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 completion.</td>
</tr>
<tr>
<td>1</td>
<td>An SQL error occurred.</td>
</tr>
<tr>
<td>2</td>
<td>The specified Web Part Page cannot be found or @SiteId, @DirName or @LeafName is NULL.</td>
</tr>
<tr>
<td>212</td>
<td>The specified Site Collection is Locked.</td>
</tr>
<tr>
<td>1816</td>
<td>The Quota for the specified Site Collection has been exceeded.</td>
</tr>
</tbody>
</table>

Result Sets: MUST NOT return any result sets.

3.1.4.3 proc_AddSolution

The proc_AddSolution stored procedure is called to add a sandboxed solution to a site collection. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_AddSolution (  
@Name                     nvarchar(128),  
@SiteId                   uniqueidentifier,  
@SolutionId               uniqueidentifier,  
@Hash                     nvarchar(50),  
@ValidatorsHash           char(64),  
@SolutionGalleryItemId    int,  
@Status                   smallint,  
@HasAssemblies            tinyint,  
@Definitions              varbinary(max)  
);  

@Name: The name of the sandboxed solution.

@SiteId: The Site Collection identifier of the Site Collection on which the sandboxed solution is to be added. This value MUST NOT be NULL.

@SolutionId: The identifier of the sandboxed solution. This value MUST NOT be NULL.

@Hash: The implementation-specific hash of the content of the sandboxed solution. This value MUST NOT be NULL.

@ValidatorsHash: An implementation-specific hash that uniquely identifies the validation programs that were run on the sandboxed solution. This value is determined by the implementation of the protocol server. This value MUST NOT be NULL.

@SolutionGalleryItemId: The list item identifier of the List Item in Solution Gallery List that contains this solution. This value MUST NOT be NULL.
@Status: This value MUST be 1.

@HasAssemblies: MUST be 1 if the sandboxed solution contains assemblies. MUST be 0 otherwise.

@Definitions: The implementation-specific serialization of the feature definitions for this sandboxed solution. This value is determined by the implementation of the protocol server.

Return values: An integer which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.1.4.4 proc_AddWebPart

The proc_AddWebPart stored procedure is called to add a Web Part to a Web Part Page. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_AddWebPart(
    @SiteId                   uniqueidentifier,
    @DirName                  nvarchar(256),
    @LeafName                 nvarchar(128),
    @Level                    tinyint OUTPUT,
    @bAllUser                 bit,
    @UserId                   int,
    @WebPartID                uniqueidentifier,
    @WebPartTypeID            uniqueidentifier,
    @Assembly                 nvarchar(255),
    @Class                    nvarchar(255),
    @SolutionId               uniqueidentifier,
    @TheListID                uniqueidentifier,
    @bCheckLock               bit,
    @IsIncluded               bit,
    @FrameState               tinyint,
    @ZoneID                   nvarchar(64),
    @PartOrder                int,
    @TheFlags                 int,
    @TheType                  tinyint,
    @TheBaseViewID            tinyint,
    @AllUsersProperties       varbinary(max),
    @PerUserProperties        varbinary(max),
    @WebPartIdProperty        nvarchar(255),
    @RequestGuid              uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The Site Collection identifier of the Site Collection which contains the Web Part Page to which to add the Web Part. This value MUST NOT be NULL.

@DirName: The Directory Name of the Web Part Page to which to add the Web Part. This value MUST NOT be NULL.

@LeafName: The Leaf Name of the Web Part Page to which to add the Web Part. This value MUST NOT be NULL.

@Level: The publishing level of the Web Part Page. The value is returned as an output parameter and MUST be the same value passed in or 2 (Draft). The value is changed to 2 (Draft) if the Web Part Page is in a Document Library, @Level is published, @bCheckLock is 1, @bAllUser is 1,
@UserId references an existing user in the Site Collection, the Web Part Page is moderated or has minor version control enabled, and creation of a new version of the Web Part Page succeeded.

@bAlluser: If this flag is set to 1 the Web Part is added to the Shared View of the Web Part Page and is available to All Users. If this flag is set to 0, @UserId is used to add the Web Part to the current user’s personal View of the Web Part Page and is available only to the current user.

@UserId: The User Identifier (as specified in [MS-WSSFO2], section 2.2.1.13) for the current user. If the Web Part Page is Moderated or has minor version control enabled then @UserId is used to track who is adding the Web Part.

@WebPartID: The Web Part Identifier (as specified in [MS-WSSFO2], section 2.2.1.15) of the Web Part being added. This value MUST NOT be NULL.

@WebPartTypeID: The Web Part type identifier of the Web Part being added. This value MUST NOT be NULL.

@Assembly: The fully qualified name of the assembly that implements the web part.

@Class: The name of the .NET class that implements the web part.

@SolutionId: The identifier of the sandboxed solution that implements the web part.

@TheListID: The list identifier of the list associated with the web part.

@bCheckLock: If this flag is set to 1, check if the document is in a state where it can be modified. If it cannot be modified, return specific return code values defined in the following Return Code Values table that explain why it cannot be modified. If this flag is set to 0, the checks made when this flag is set to 1 are bypassed.

@IsIncluded: The Web Part Is Closed State of the Web Part.

@FrameState: The Web Part Chrome State of the Web Part.

@ZoneID: The name of the Web Part Zone identifier of the Web Part Zone to which to add the Web Part.

@PartOrder: The Web Part Zone Index of the Web Part.

@TheFlags: The View Flags, as specified [MS-WSSFO2] section 2.2.2.12, of the Web Part.

@TheType: The web part type identifier of the web part.

@TheBaseViewID: The base view identifier of the web part.

@AllUsersProperties: A binary payload containing 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.

@PerUserProperties: A binary payload containing zero or more personalizable properties on the Web Part. If this value is NULL then default values will be used for all of the personalizable properties on the Web Part.

@WebPartIdProperty: The HTML (HyperText Markup Language) ID attribute of the Web Part. May be NULL. If not NULL, it MUST be unique per Web Part Page.

@RequestGuid: The optional request identifier for the current request.
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 completion.</td>
</tr>
<tr>
<td>1</td>
<td>An error occurred executing the stored procedure.</td>
</tr>
<tr>
<td>2</td>
<td>The Web Part Page cannot be found.</td>
</tr>
<tr>
<td>3</td>
<td>The Web Part Page is in a Document Library, @Level is Published, @bCheckLock is 1, @bAllUser is 1, @UserId references an existing user in the Site Collection, the Web Part Page is Moderated or has minor version control enabled, and a new Draft version of the Web Part Page cannot be created because a unique name for it cannot be created.</td>
</tr>
<tr>
<td>12</td>
<td>@bCheckLock is 1, @bAllUser is 0 and the Web Part Page is Checked Out.</td>
</tr>
<tr>
<td>33</td>
<td>@bCheckLock is 1, @bAllUser is 1, and the specified Web Part Page is not the current version.</td>
</tr>
<tr>
<td>87</td>
<td>The Web Part Page is in a Document Library, @Level is Published, @bCheckLock is 1, @bAllUser is 1, @UserId references an existing user in the Site Collection, the Web Part Page is Moderated or has minor version control enabled, and a new Draft version of the Web Part Page cannot be created.</td>
</tr>
<tr>
<td>158</td>
<td>@bCheckLock is 1, @bAllUser is 1, the Web Part Page is in a Document Library with Required Checkout set and it is not Checked Out.</td>
</tr>
<tr>
<td>160</td>
<td>The Web Part Page is in a Document Library, @Level is Published, @bCheckLock is 1, @bAllUser is 1, the Web Part Page is Moderated or has minor version control enabled, and @UserId is NULL.</td>
</tr>
<tr>
<td>212</td>
<td>The Site Collection is Locked.</td>
</tr>
<tr>
<td>1816</td>
<td>The Quota for the Site Collection has been exceeded.</td>
</tr>
</tbody>
</table>

Result Sets: MUST NOT return any result sets.

3.1.4.5  proc_AddWorkflow

The proc_AddWorkflow stored procedure is called to create a Workflow and add it to a list Item. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE dbo.proc_AddWorkflow (  @WorkflowTemplateId       uniqueidentifier,  @WorkflowInstanceId       uniqueidentifier,  @SiteId                   uniqueidentifier,  @WebId                    uniqueidentifier,  @ListId                   uniqueidentifier,  @(itemId)                  int,  @Level                    tinyint,  @Version                   int,  @TaskListId                uniqueidentifier,  @AdminTaskListId          uniqueidentifier,  @Author                   int,  @ProcessingId             uniqueidentifier,  @InstanceData              varbinary(max),  @InstanceDataSize         int,  @Modifications            nvarchar(max),  @StatusFieldOrdinal       int,  @StatusField             nvarchar(64),
```
@RequestGuid              uniquidentifier = NULL OUTPUT
);

@WorkflowTemplateId: The Workflow Template identifier of the workflow template on which the Workflow being added is based.

@WorkflowInstanceId: The Workflow identifier of the Workflow being added, or NULL. There MUST NOT be an existing workflow with the same workflow identifier. If @WorkflowInstanceId is NULL, the server MUST create a new identifier for the workflow. The server MUST set the creation and modification dates and times of the Workflow to the date and time in UTC the stored procedure is called.

@SiteId: The Site Collection identifier of the Site Collection which contains the Workflow. This value MUST NOT be NULL.

@WebId: The Site Identifier, as specified in [MS-WSSFO2], section 2.2.1.11, of the Site which contains the Workflow. This value MUST NOT be NULL.

@ListId: The list identifier of the list which contains the Workflow. This value MUST NOT be NULL.

@ItemID: The List Item Identifier (as specified in [MS-WSSFO2], section 2.2.1.6) of the list Item the Workflow is created for. This value MUST NOT be NULL.

@Level: If @ListId represents a Document Library, this represents the publishing level of the Document corresponding to the list Item specified by @ItemID. If @ListId does not represent a Document Library, this MUST be 1.

@Version: This parameter MUST contain the list item version of the list Item or the value 0. If the value is 0, the protocol server MUST ignore the list item version. If the value is nonzero and does not match the current list item version value of the list item, the protocol server MUST NOT add the workflow.

@TaskListId: The list identifier of the Workflow Task list of the workflow. This value MUST NOT be NULL.

@AdminTaskListId: This parameter MUST be NULL.

@Author: The User identifier of the user creating the Workflow. This value MUST NOT be NULL.

@ProcessingId: This parameter MUST be NULL.

@InstanceData: This parameter MUST be NULL.

@InstanceDataSize: This parameter MUST contain the value 0.

@Modifications: This parameter MUST contain an empty string.

@StatusFieldOrdinal: The ordinal of the Workflow Status field of the Workflow.
@StatusFieldOrdinal MUST be NULL if and only if @StatusField is NULL.

@StatusField: The name of the Workflow Status field of the workflow. The server MUST update the field specified by @StatusField and @StatusFieldOrdinal of the list Item specified by @ItemID to the Workflow identifier of the Workflow.

@RequestGuid: The optional request identifier for the current request.

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 completion.</td>
</tr>
<tr>
<td>2</td>
<td>No workflow association was found based on the workflow template specified by @WorkflowTemplateId in the site collection specified by @SiteId.</td>
</tr>
<tr>
<td>82</td>
<td>The workflow could not be added.</td>
</tr>
<tr>
<td>87</td>
<td>At least one input parameter was invalid.</td>
</tr>
<tr>
<td>183</td>
<td>The list item specified by @ItemId already has a workflow that is not a completed workflow.</td>
</tr>
<tr>
<td>1150</td>
<td>The list item version of the list item specified by @ItemId does not match the value of @Version.</td>
</tr>
</tbody>
</table>

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

### 3.1.4.6 proc_AddWorkflowAssociation

The `proc_AddWorkflowAssociation` stored procedure is called to add a workflow association. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE dbo.proc_AddWorkflowAssociation(
    @Id uniqueidentifier,
    @BaseId uniqueidentifier,
    @ParentId uniqueidentifier,
    @Name nvarchar(255),
    @Description nvarchar(1023),
    @StatusFieldName nvarchar(64),
    @SiteId uniqueidentifier,
    @WebId varbinary(16),
    @ListId varbinary(16),
    @ContentTypeId varbinary(512),
    @TaskListId varbinary(16),
    @HistoryListId varbinary(16),
    @TaskListTitle nvarchar(255),
    @HistoryListTitle nvarchar(255),
    @Author int,
    @Configuration int,
    @AutoCleanupDays int,
    @PermissionsManual bigint,
    @InstantiationParams nvarchar(max),
    @RequestGuid uniqueidentifier = NULL OUTPUT
);
```

**@Id:** The Workflow association identifier of the Workflow association being created. If this value is NULL, the server MUST create a new Workflow association identifier for the Workflow association. The server MUST set the creation and modification times for the workflow association to the date and time in UTC when the procedure was called.

**@BaseId:** The Workflow Template identifier of the workflow template on which the Workflow association is based. This value MUST NOT be NULL.

**@ParentId:** The Workflow association identifier of the parent Workflow association of the Workflow association.

**@Name:** The name of the Workflow association.
@Description: The description of the Workflow association.

@StatusFieldName: The name of the Workflow Status field of the Workflow association.

@SiteId: The Site Collection identifier of the Site Collection which contains the Workflow association. This value MUST NOT be NULL.

@WebId: The Site Identifier (as specified in [MS-WSSFO2], section 2.2.11) of the Site which contains the Workflow association. This value MUST NOT be NULL.

@ListId: The list identifier of the list with which the Workflow association is associated.

@ContentTypeId: The Content Type Identifier (as specified in [MS-WSSFO2], section 2.2.1.1) of the Content type with which the Workflow is associated.

@TaskListId: The list identifier of the Workflow Task list of the Workflow association.

@HistoryListId: The list identifier of the Workflow History list of the Workflow association.

@TaskListTitle: The title of the Workflow Task list of the Workflow association.

@HistoryListTitle: The title of the Workflow History list of the Workflow association.

@Author: The User identifier of the Author of the Workflow association.

@Configuration: The Workflow association Configuration of the Workflow association.

@AutoCleanupDays: The number of days before Workflows based on the Workflow association are cleaned up. MUST contain a positive integer.

@PermissionsManual: The WSS Rights Mask (as specified in [MS-WSSFO2], section 2.2.2.14) required to manually start any Workflows created from the Workflow association.

@InstantiationParams: The workflow association data of the Workflow association.

@RequestGuid: The optional request identifier for the current request.

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 completion.</td>
</tr>
<tr>
<td>82</td>
<td>The workflow association was not created.</td>
</tr>
</tbody>
</table>

Result Sets: MUST NOT return any result sets.

3.1.4.7 proc_AddWorkItem

The proc_AddWorkItem stored procedure is called to add a new Work Item to the set of pending work items. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_AddWorkItem(
    @WorkItemId                 uniqueidentifier,
    @DeliveryDate               datetime,
    @Type                       uniqueidentifier,
    @SubType                    uniqueidentifier,
    @SiteId                     uniqueidentifier,
)
```
@ParentId              varbinary(16),
@ItemGuid              varbinary(16),
@BatchId               varbinary(16),
@UserId                int,
@BinaryPayload         varbinary(max),
@TextPayload           nvarchar(max),
@ProcessingId          uniqueidentifier,
@AutoDeleteOld         bit = 0,
@ExponentialRetryBackOff bit = 1,
@RequestGuid           uniqu

@WorkItemId: The Work Item identifier of the Work Item. If the parameter is not NULL, then the server MUST give the new Work Item a Work Item identifier equal to the value of the parameter. If the parameter is NULL, then the server MUST generate a GUID for the Work Item identifier.

@DeliveryDate: The Work Item Delivery Date. If the parameter is NULL, then the server MUST schedule the Work Item to run immediately. If the parameter is not NULL, then the server MUST schedule the Work Item to run on the given Delivery Date.

@Type: The Work Item type identifier of the Work Item type. MUST NOT be NULL.

@SubType: The Work Item Subtype identifier of the Work Item Subtype or, when there is no associated Work Item Subtype, the empty GUID.

@SiteId: The Site Collection identifier of the Site Collection or, when there is no associated Site Collection, the empty GUID. MUST NOT be NULL.

@ParentId: The Work Item Parent identifier of the Work Item. MUST NOT be NULL.

@ItemGuid: An Item identifier for an list item associated with the work item. SHOULD <2> be 0 if there is no associated item. MUST NOT be NULL.

@WebId: The Site Identifier (as specified in [MS-WSSFO2], section 2.2.1.11) of the Site.

@ItemGuid: The Item GUID of the list Item or, when there is no associated list Item, the empty GUID.

@BatchId: The Work Item Batch identifier of the Work Item Batch of the Work Item or the empty GUID.

@ProcessingId: The identifier of the work item process. If this parameter is NULL, then the client MUST NOT process the Work Item.

@AutoDeleteOld: This parameter indicates whether calls to proc_RevertInProgressWorkItem or proc_RevertInProgressWorkItems MUST cause the server to delete this Work Item if it has a Work Item Delivery Date 10 or more days prior to the call. A value of 0 indicates that the deletion MUST NOT occur. A value of 1 indicates that the deletion MUST occur.
@ExponentialRetryBackOff: This parameter indicates whether or not the server MUST retry execution with exponential backoff from the Work Item Delivery Date when the client calls proc_RevertInProgressWorkItem or proc_RevertInProgressWorkItems. A value of 0 indicates that exponential backoff MUST NOT occur on retry. A value of 1 indicates that exponential backoff MUST occur on retry.

@RequestGuid: The optional request identifier for the current request.

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 completion.</td>
</tr>
<tr>
<td>5</td>
<td>Error: Access denied.</td>
</tr>
<tr>
<td>82</td>
<td>Error: Failed to add the Work Item.</td>
</tr>
</tbody>
</table>

Result Sets: MUST NOT return any result sets.

3.1.4.8 proc_ApplyViewToListWebPart

The proc_ApplyViewToListWebPart stored procedure is called to apply the specified View to the specified list View Web Part. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_ApplyViewToListWebPart(
    @SiteId                   uniqueidentifier,
    @DirName                  nvarchar(256),
    @LeafName                 nvarchar(128),
    @Level                    tinyint OUTPUT,
    @WebPartID                uniqueidentifier,
    @ViewId                   uniqueidentifier,
    @UserId                   int,
    @ViewEditPerms            int,
    @ViewBody                 nvarchar(max),
    @ViewFlags                int OUTPUT,
    @BaseViewId               int OUTPUT,
    @RequestGuid              uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The Site Collection identifier of the Site Collection which contains the Web Part specified by @WebPartID. MUST NOT be NULL.

@DirName: The Directory Name of the Web Part Page that contains the Web Part specified by @WebPartID. MUST NOT be NULL.

@LeafName: The Leaf Name of the Web Part Page that contains the Web Part specified by @WebPartID. MUST NOT be NULL.

@Level: This is an input/output parameter. On input, this is the Publishing Level value of the Page specified by @LeafName that contains the Web Part specified by @WebPartID. On output, this is the Publishing Level of the Page specified by @LeafName after the specified View has been applied to the Web Part specified by @WebPartID. MUST NOT be NULL.

@WebPartID: The Web Part Identifier (as specified in [MS-WSSFO2], section 2.2.1.15) of the Web Part on which to apply the specified View. MUST NOT be NULL.

[MS-WSSPROG2] — v20120630
Windows SharePoint Services: Content Database Programmability Extensions Communications Version 2 Protocol Specification

Copyright © 2012 Microsoft Corporation.

Release: July 16, 2012
@ViewId: If @ViewId is not NULL, it is the GUID for a list View Web Part. The base view identifier, Content type Identifier (as specified in [MS-WSSFO2], section 2.2.1.1), and View Flags (as specified in [MS-WSSFO2], section 2.2.2.12) from the Web Part specified by @ViewId MUST be copied to the Web Part specified by @WebPartID. If @ViewId is NULL, the base view identifier MUST be set to 0 on the Web Part specified by @WebPartID. The View Flags (as specified in [MS-WSSFO2], section 2.2.2.12) MUST be copied from @ViewFlags to the Web Part specified by @WebPartID. The Content Type Identifier (as specified in [MS-WSSFO2], section 2.2.1.1) of the Web Part specified by @WebPartID MUST NOT be changed.

@UserId: The User Identifier (as specified in [MS-WSSFO2], section 2.2.1.13) for the user that is applying the specified View to the Web Part specified by @WebPartID. MUST NOT be NULL.

@ViewEditPerms: The set of permission flags for the User specified by @UserId. MUST NOT be NULL. @ViewEditPerms MUST be a bitwise logical combination of the values listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00000001</td>
<td>Add a personal View to a list view page</td>
</tr>
<tr>
<td>0x00000002</td>
<td>Add a personal View to a Web Part Page other than a list View Page</td>
</tr>
<tr>
<td>0x00000004</td>
<td>Add a public view to a list View Page</td>
</tr>
<tr>
<td>0x00000008</td>
<td>Add a Public View to a Web Part Page other than a list View Page</td>
</tr>
<tr>
<td>0x00000010</td>
<td>Modify a personal View on a list View Page</td>
</tr>
<tr>
<td>0x00000020</td>
<td>Modify a personal View on a Web Part Page other than a list View Page</td>
</tr>
<tr>
<td>0x00000040</td>
<td>Modify a Public View on a list View Page</td>
</tr>
<tr>
<td>0x00000080</td>
<td>Modify a Public View on a Web Part Page other than a list View Page</td>
</tr>
</tbody>
</table>

@ViewBody: The CAML for the View to be applied to the Web Part specified by @WebPartID.

@ViewFlags: This is an input/output parameter. On input, if @ViewId is NULL, this set of View Flags (as specified in [MS-WSSFO2], section 2.2.2.12) MUST be applied to the Web Part specified by @WebPartID. The VIEWFLAG_HIDDEN (0x00000008) bit MUST be set. The VIEWFLAG_PERSONAL (0x00040000) bit MUST be set if the Web Part is a personal Web Part or cleared otherwise. On input, if @ViewId is not NULL, @ViewFlags MUST be ignored. On output, this is the set of View Flags (as specified in [MS-WSSFO2], section 2.2.2.12) of the Web Part specified by @WebPartID after the specified View has been applied.

@BaseViewId: This is an output parameter. On output, this is the base view identifier for this Web Part.

@RequestGuid: The optional request identifier for the current request.

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 completion.</td>
</tr>
<tr>
<td>1</td>
<td>Internal SQL error.</td>
</tr>
<tr>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>2</td>
<td>The Page specified by @DirName, @LeafName, and @Level does not exist or has been deleted; or the Web Part specified by @WebPartID does not exist or has been deleted.</td>
</tr>
<tr>
<td>3</td>
<td>The Web Part Page is Moderated or has minor version control enabled, and a new version of the Web Part Page cannot be created because a unique name for it cannot be created.</td>
</tr>
<tr>
<td>5</td>
<td>The user specified by @UserId does not have the necessary Permissions to modify the Web Part specified by @WebPartID.</td>
</tr>
<tr>
<td>12</td>
<td>Cannot modify a personal Web Part on a Page that is Checked Out.</td>
</tr>
<tr>
<td>33</td>
<td>The Page specified by @DirName, @LeafName, and @Level is not the Current Version.</td>
</tr>
<tr>
<td>87</td>
<td>The Page specified by @DirName, @LeafName, and @Level does not exist or has been deleted.</td>
</tr>
<tr>
<td>158</td>
<td>The Page specified by @DirName and @LeafName needs to be Checked Out because the Page lives in a Document Library with Required Checkout set.</td>
</tr>
<tr>
<td>160</td>
<td>Need to create a new version of the Page specified by @DirName and @LeafName, but no user is specified by @UserId.</td>
</tr>
<tr>
<td>212</td>
<td>Need to create a new version of the Page specified by @DirName and @LeafName, but the Site Collection specified by @SiteId is Locked.</td>
</tr>
<tr>
<td>1816</td>
<td>Need to create new version of Page specified by @DirName and @LeafName, but the Site Collection specified by @SiteId has exceeded its Quota.</td>
</tr>
<tr>
<td>-2147467259</td>
<td>An error occurred while the stored procedure was running.</td>
</tr>
</tbody>
</table>

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

### 3.1.4.9 proc_AutoCleanupWorkflows

The proc_AutoCleanupWorkflows stored procedure is called to clean up completed workflows. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_AutoCleanupWorkflows(
    @RequestGuid uniqueidentifier = NULL OUTPUT
);
```

@RequestGuid: The optional request identifier for the current request.

**Return Code Values:** This stored procedure returns an integer Return Code which the client MUST ignore.

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

### 3.1.4.10 proc_AutoDropWorkflows

The proc_AutoDropWorkflows stored procedure is called to delete workflows and workflow associations. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE dbo.proc_AutoDropWorkflows (
```
@SiteId: The Site Collection identifier of the Site Collection which contains the Workflows and Workflow associations. This value MUST NOT be NULL. The server MUST update the Site Collection Quota (3.1.1.5) to remove the space used by the deleted Workflows.

@WebId: The Site Identifier (as specified in [MS-WSSFO2], section 2.2.1.11) of the Site which contains the Workflows and Workflow associations. If this value is NULL, the server MUST include all Sites.

@ListId: The list identifier of the list that is associated with the Workflows and Workflow associations. If this value is NULL, the server MUST include all lists.

@ListItemId: The List Item Identifier (as specified in [MS-WSSFO2], section 2.2.1.6) of the list for which the Workflows were created. If @ListId is NULL, this value MUST be NULL. If this value is NULL, the server MUST include all list Items.

@TemplateId: The Workflow Template identifier of the workflow template of the Workflow associations. If this value is NULL, the server MUST include all workflow templates.

@AutoCleanupDate: The date and time limit for deleting Workflow associations and Workflows. If @ForceDelete contains the value 1, the server MUST ignore @AutoCleanupDate. If @ForceDelete contains the value 0, @AutoCleanupDate MUST contain a valid date value that occurs before the date and time that proc_AutoDropWorkflows was called.

@ForceDelete: This parameter determines whether to delete all Workflows or only those that are complete and were last modified before @AutoCleanupDate. This value MUST be 0 or 1. When @ForceDelete is 0, the server MUST delete only Workflows that are complete and were last modified before the date specified by @AutoCleanupDate. When @ForceDelete is 1, the server MUST delete all workflows up to the @TopBeforeQuick limit meeting the criteria specified by @SiteId, @WebId, @ListId, @ListItemId and @TemplateId, and MUST ignore completion and modification date.

@TopBeforeQuick: This parameter limits the number of Workflows being deleted. This value MUST contain a positive integer or 0. The server MUST NOT delete more workflows than the number specified by @TopBeforeQuick. When @ForceDelete is 1, the @TopBeforeQuick limit is reached, and @TemplateId is not null, the server MUST mark all Workflow associations in the Site Collection specified by @SiteId and based on the workflow template specified by @TemplateId for deferred deletion by proc_AutoCleanupWorkflows. When @ForceDelete is 1, the @TopBeforeQuick limit is reached, and @TemplateId is null, the server MUST mark all workflows that were not deleted and that meet the criteria specified by the @SiteId, @WebId, @ListId and @ListItemId and @TemplateId parameters for deferred deletion by proc_AutoCleanupWorkflows.

@RequestGuid: The optional request identifier for the current request.

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 completion; <code>@ForceDelete</code> is 0, or <code>@ForceDelete</code> is 1 and some of the workflows meeting the criteria specified by the input parameters were not deleted because of the <code>@TopBeforeQuick</code> limit.</td>
</tr>
<tr>
<td>1</td>
<td>Successful completion; <code>@ForceDelete</code> is 1, and all workflows meeting the criteria specified by the input parameters were deleted.</td>
</tr>
</tbody>
</table>

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

### 3.1.4.11 proc_CancelDeclarativeWorkflows

The `proc_CancelDeclarativeWorkflows` stored procedure is called to cancel all declarative workflows in a site collection. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_CancelDeclarativeWorkflows(  
    @SiteId                  uniqueidentifier,  
    @RequestGuild            uniqueidentifier = NULL OUTPUT  
);  
```

- **@SiteId:** The Site Collection identifier of the Site Collection which contains the Workflows.
- **@RequestGuid:** The optional request identifier for the current request.

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

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

### 3.1.4.12 proc_CancelWorkflow

The `proc_CancelWorkflow` stored procedure is called to cancel a workflow. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_CancelWorkflow(  
    @SiteId                  uniqueidentifier,  
    @WorkflowInstanceId      uniqueidentifier,  
    @RequestGuid             uniqueidentifier = NULL OUTPUT  
);  
```

- **@SiteId:** The Site Collection identifier of the Site Collection which contains the Workflow.
- **@WorkflowInstanceId:** The Workflow identifier of the Workflow being canceled. If the Workflow is completed or terminated, the stored procedure MUST NOT update the Workflow. The server MUST add the `WFS_CANCELED` flag to the workflow internal state and MUST remove the `WFS_RUNNING`, `WFS_LOCKED` and `WFS_HASNEWEVENTS` flags. If the `Workflow_Status1` field of the Workflow is `WFSTAT_FAILEDTOSTART_RETRY`, the server MUST set the Workflow Status1 field to `WFSTAT_FAILEDTOSTART`; otherwise, the server MUST set the Workflow Status1 field to `WFSTAT_CANCELED`. The server MUST delete any list Items in the Workflow Task list for the Workflow and any work items scheduled to process the Workflow. The server MUST set the modification date and time of the workflow to the date and time in UTC when the procedure was called.
- **@RequestGuid:** The optional request identifier for the current request.
Return Code Values: An integer which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.1.4.13 proc_CommitUpdatedZoneIds

The proc_CommitUpdatedZoneIds stored procedure is called to update the webpart zone identifiers of webparts contained in a webpart page. The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_CommitUpdatedZoneIds (  
    @SiteId uniqueidentifier,  
    @DocId uniqueidentifier,  
    @ZoneIdNew nvarchar(64),  
    @ZoneIdOld nvarchar(64)  
);  
```

@SiteId: The Site Collection identifier of the Site Collection which contains the Web Parts to be modified. MUST NOT be NULL.

@DocId: The Document Identifier (as specified in [MS-WSSFO2] section 2.2.1.2) of the Document which contains the Web Parts to be modified. MUST NOT be NULL.

@ZoneIdNew: The webpart zone identifier of the webpart zone containing the web parts to be updated. MUST NOT be NULL.

@ZoneIdOld: The webpart zone identifier to be replaced. MUST NOT be NULL.

Return values: An integer which the protocol client MUST ignore.

Result Sets: MUST NOT return any result sets.

3.1.4.14 proc_CompleteInProgressWorkItems

The proc_CompleteInProgressWorkItems stored procedure is called to mark a set of Work Items as Completed Work Items. The server MUST restrict the set to those specified by the parameters and for which the Work Item Delivery Date has passed. The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_CompleteInProgressWorkItems(  
    @ProcessingId uniqueidentifier,  
    @SiteId uniqueidentifier,  
    @ParentId uniqueidentifier,  
    @WorkItemType uniqueidentifier,  
    @BatchId uniqueidentifier,  
    @RequestGuid uniqueidentifier = NULL OUTPUT  
);  
```

@ProcessingId: The Work Item Process identifier of the Work Item Process. The server MUST only modify Work Items associated with this Work Item Process.

@SiteId: The Site Collection identifier of the Site Collection. If @SiteId is not NULL, then the server MUST only modify Work Items associated with this Site Collection. If @SiteId is NULL, then the server MUST modify Work Items that meet the criteria specified by the other parameters regardless of associated Site Collection identifier.
@ParentId: The Work Item Parent identifier of the Work Item. If @ParentId is not NULL, then the server MUST only modify Work Items which have this Work Item Parent identifier. If @ParentId is NULL, then the server MUST modify Work Items that meet the criteria specified by the other parameters regardless of the value of their Work Item Parent identifier.

@WorkItemType: The Work Item type identifier of the Work Item type. The server MUST only modify Work Items associated with this Work Item type. MUST NOT be NULL.

@BatchId: The Work Item Batch identifier of the Work Item Batch. If @BatchId is not NULL, then the server MUST only modify Work Items associated with this Work Item Batch. If @BatchId is NULL, then the server MUST modify Work Items that meet the criteria specified by the other parameters regardless of associated Work Item Batch identifier.

@RequestGuid: The optional request identifier for the current request.

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

Result Sets: MUST NOT return any result sets.

3.1.4.15 proc_CopyDefaultViewWebParts

The proc_CopyDefaultViewWebParts stored procedure is called to copy Web Parts from the Shared View of a Web Part Page to a new Web Part Page. Personal View Web Parts and the default list view Web Part are skipped. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_CopyDefaultViewWebParts(
    @SiteID                   uniqueidentifier,
    @DefaultViewDirName       nvarchar(256),
    @DefaultViewLeafName      nvarchar(128),
    @NewViewDocId             uniqueidentifier,
    @RequestGuid              uniqueidentifier = NULL OUTPUT
);
```

@ SiteID: The Site Collection identifier of the Site Collection which contains the source and destination Web Part Pages. This parameter MUST NOT be NULL.

@ DefaultViewDirName: The Directory Name of the source Web Part Page. This parameter MUST NOT be NULL.

@ DefaultViewLeafName: The Leaf Name of the source Web Part Page. This parameter MUST NOT be NULL.

@ NewViewDocId: The GUID of the Web Part Page where the copied Web Parts will be placed. This parameter MUST NOT be NULL.

@RequestGuid: The optional request identifier for the current request.

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 completion.</td>
</tr>
<tr>
<td>2</td>
<td>The system cannot find the Web Part Page specified.</td>
</tr>
<tr>
<td>212</td>
<td>The specified Site Collection is Locked.</td>
</tr>
</tbody>
</table>

[MS-WSSPROG2] — v20120630
Windows SharePoint Services: Content Database Programmability Extensions Communications Version 2 Protocol Specification

Copyright © 2012 Microsoft Corporation.

Release: July 16, 2012
<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1150</td>
<td>A concurrency violation occurred. No such version of the Web Part Page exists.</td>
</tr>
<tr>
<td>1816</td>
<td>The Quota for the specified Site Collection has been exceeded.</td>
</tr>
<tr>
<td>-2147467259</td>
<td>An error occurred while the stored procedure was running.</td>
</tr>
</tbody>
</table>

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

### 3.1.4.16 proc_CountWorkflowAssociations

The proc_CountWorkflowAssociations stored procedure is called to obtain a count of workflow associations for one or all workflow templates contained in a site collection. The T-SQL syntax for the stored procedure is as follows:

```tsql
PROCEDURE proc_CountWorkflowAssociations(
    @SiteId uniqueidentifier,
    @BaseId uniqueidentifier,
    @RequestGuid uniqueidentifier = NULL OUTPUT );
```

- **@SiteId**: The Site Collection identifier of the Site Collection which contains the Workflow associations.
- **@BaseId**: The Workflow Template identifier of the workflow template the Workflow associations are based on. If this value is NULL, the server MUST include all workflow templates.
- **@RequestGuid**: The optional request identifier for the current request.

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

**Result Sets**: MUST return the following result set:

#### 3.1.4.16.1 Count Workflow Associations Result Set

This Result Set contains exactly one row. The T-SQL syntax for the result set is as follows:

```tsql
{Count}    int;
```

- **{Count}**: The count of the workflow associations meeting the criteria specified by the input parameters.

### 3.1.4.17 proc_CountWorkflows

The proc_CountWorkflows stored procedure is called to obtain a count of workflows based on a workflow association or workflow template. The T-SQL syntax for the stored procedure is as follows:

```tsql
PROCEDURE proc_CountWorkflows(
    @AssociationId uniqueidentifier,
    @SiteId uniqueidentifier,
    @BaseId uniqueidentifier,
    @InternalState int,
    @RequestGuid uniqueidentifier = NULL OUTPUT
 );
```
@AssociationId: The Workflow association identifier of the Workflow association of the Workflows. If this value is not NULL, the server MUST ignore @SiteId and @BaseId. If this value is NULL, the server MUST include all Workflow associations.

@SiteId: The Site Collection identifier of the Site Collection which contains the Workflows.

@BaseId: The Workflow Template identifier of the workflow template the Workflows are based on.

@InternalState: A workflow internal state bitmask specifying the internal states of the Workflows. If @InternalState is not NULL, the server MUST restrict the count in the result set to Workflows which have at least one internal state flag in common with the bitmask, similar to Workflow.InternalState & @InternalState <> 0.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST return the following result set:

3.1.4.17.1 Count Workflows Result Set
This Result Set contains exactly one row. The T-SQL syntax for the result set is as follows:

```
{Count} int;
```

{Count}: The count of the workflows meeting the criteria specified by the input parameters.

3.1.4.18 proc_CountWorkflowsBatch
The proc_CountWorkflowsBatch stored procedure is called to obtain a set of workflow templates and the count of workflows based on each workflow template. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE dbo.proc_CountWorkflowsBatch(
    @SiteId uniqueidentifier,
    @WebId uniqueidentifier,
    @ListId uniqueidentifier,
    @ContentTypeId uniqueidentifier,
    @InternalState int,
    @RequestGuid uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The Site Collection identifier of the Site Collection which contains the Workflows.

@WebId: The Site Identifier (as specified in [MS-WSSFO2], section 2.2.1.11) of the Site which contains the Workflows.

@ListId: The list identifier of the list which contains the Workflows. If @ListId is NULL, the server MUST include all lists.
**@ContentTypeId:** The Content Type identifier (as specified in [MS-WSSFO2], section 2.2.1.1) of the Content type from which the Workflows were created. If @ListId is not NULL, the server MUST ignore @ContentTypeId.

**@InternalState:** A workflow internal state bitmask specifying the internal states of the Workflows. If @InternalState is not NULL, the server MUST restrict the result to workflows which have an internal state that has at least one internal state flag in common with the bitmask (that is, Workflow.InternalState & @InternalState <> 0).

**@RequestGuid:** The optional request identifier for the current request.

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

**Result Sets:** MUST return the following result set:

### 3.1.4.18.1 Workflows Batch Result Set

This Result Set returns a set of workflow template identifiers and the count of workflows based on each template. The T-SQL syntax for the result set is as follows:

```
TemplateId             uniqueidentifier
{Count}                int;
```

**TemplateId:** The Workflow Template identifier. At least one Workflow specified by the input parameters MUST be based on the workflow template.

**{Count}:** The count of the Workflows based on the workflow template. This value MUST be greater than zero.

### 3.1.4.19 proc_CreateListViewPart

The proc_CreateListViewPart stored procedure is called to add a list View Web Part to a Web Part Page. The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_CreateListViewPart(
    @SiteId                   uniqueidentifier,
    @bAllUser                 bit,
    @UserId                   int,
    @DirName                  nvarchar(256),
    @LeafName                 nvarchar(128),
    @Level                    tinyint OUTPUT,
    @WebPartID                uniqueidentifier,
    @WebPartTypeID            uniqueidentifier,
    @IsIncluded               bit,
    @FrameState               tinyint,
    @ZoneID                   nvarchar(64),
    @PartOrder                int,
    @ListId                   uniqueidentifier,
    @BaseViewId               uniqueidentifier,
    @Flags                    int,
    @ContentTypeId            tContentTypeId,
    @AllUsersProperties       varbinary(max),
    @PerUserProperties        varbinary(max),
    @WebPartIdProperty        nvarchar(255),
    @View                     varbinary(max),
    @DisplayName              nvarchar(255),
```

52 / 157
@RequestGuid              uniqueidentifier = NULL OUTPUT
);

@SiteId: The Site Collection identifier of the Site Collection which contains the Web Part Page to which the list View Web Part will be added. MUST NOT be NULL.

@bAllUser: Specifies whether to add the Web Part for the Shared View or personal View of the Web Part Page. If this flag is set to 1 the Web Part is added to the Shared View of the Web Part Page and is available to All Users. If this flag is set to 0 @UserId is used to add the Web Part to the current user's personal View of the Web Part Page and is available only to the current user. MUST NOT be NULL.

@UserId: The User Identifier (as specified in [MS-WSSFO2], section 2.2.1.13) for the current user. If the Web Part Page is moderated or has version control enabled then @UserId is used to track who is adding the Web Part.

@DirName: The Directory Name of the Web Part Page to which to add the list View Web Part. MUST NOT be NULL.

@LeafName: The Leaf Name of the Web Part Page to which to add the list View Web Part. MUST NOT be NULL.

@Level: The Publishing Level of the Web Part Page for the current user. The value is returned as an output parameter and MUST be the same value passed in or 2 (Draft). The value is changed to 2 (Draft) if the Web Part Page is in a Document Library, @Level is 1 (Published), @bCheckLock is 1, @bAllUser is 1, @UserId references an existing user in the Site Collection, the Web Part Page is Moderated or has Version Control enabled, and creation of a new version of the Web Part Page succeeded.

@WebPartID: The Web Part identifier (as specified in [MS-WSSFO2], section 2.2.1.14) of the Web Part being added. MUST NOT be NULL.

@WebPartTypeID: The Web Part type identifier of the Web Part being added. MUST NOT be NULL.

@IsIncluded: The Web Part Is Closed State of the added Web Part.

@FrameState: The Web Part chrome state of the added Web Part.

@ZoneID: The Web Part Zone identifier of the Web Part Zone to which to add the Web Part.

@PartOrder: The Web Part Zone Index of the added Web Part.

@ListId: The List identifier (as specified in [MS-WSSFO2], section 2.2.1.5) of the list for the Web Part

@BaseViewId: The base view identifier for the Web Part.

@Flags: A View Flags (as specified in [MS-WSSFO2], section 2.2.2.12) value specifying View related settings for the Web Part.

@ContentTypeId: The Content type identifier (as specified in [MS-WSSFO2], section 2.2.1.1) of the list Items in the list to be displayed in the Web Part.

@AllUsersProperties: A binary payload containing zero or more customizable properties on the Web Part.
@PerUserProperties: A binary payload containing zero or more personalizable properties on the Web Part.

@WebPartIdProperty: The HTML (HyperText Markup Language) ID attribute of the Web Part. May be NULL. If not NULL, it MUST be unique per Web Part Page.

@View: CAML XML specifying View related settings for the Web Part.

@DisplayName: The Display Name for the Web Part.

@RequestGuid: The optional request identifier for the current request.

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 completion.</td>
</tr>
<tr>
<td>1</td>
<td>The List View Web Part was not successfully created.</td>
</tr>
<tr>
<td>2</td>
<td>The specified Web Part Page cannot be found.</td>
</tr>
<tr>
<td>3</td>
<td>The Web Part Page is Moderated or has minor version control enabled, and a new version of the Web Part Page cannot be created.</td>
</tr>
<tr>
<td>12</td>
<td>Attempted to add a personalized list View Web Part to a Web Part Page whose Publishing Level is Checked Out.</td>
</tr>
<tr>
<td>33</td>
<td>The specified Web Part Page is not the Current Version.</td>
</tr>
<tr>
<td>87</td>
<td>The Web Part Page is in a Document Library, @Level is 1 (Published), @bAllUser is 1, @UserId references an existing user in the Site Collection, the Web Part Page is Moderated or has minor version control enabled, and a new Draft version of the Web Part Page cannot be created.</td>
</tr>
<tr>
<td>158</td>
<td>The Web Part Page is required to be Checked Out before it is modified and it is not Checked Out.</td>
</tr>
<tr>
<td>160</td>
<td>The Web Part Page is in a Document Library, @Level is 1 (Published), @bAllUser is 1, the Web Part Page is Moderated or has minor version control enabled, and @UserId is NULL.</td>
</tr>
<tr>
<td>212</td>
<td>The specified Site Collection has been Locked.</td>
</tr>
<tr>
<td>1816</td>
<td>The Quota for the specified Site Collection has been exceeded.</td>
</tr>
</tbody>
</table>

Result Sets: MUST NOT return any result sets.

3.1.4.20 proc_DeleteDocEventReceiver

The proc_DeleteDocEventReceiver stored procedure is called to delete the registration of an event receiver for a specified document. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_DeleteDocEventReceiver(
    @DocUrl                   nvarchar(260),
    @Id                       uniqueidentifier,
    @Name                     nvarchar(256),
    @SiteId                   uniqueidentifier,
    @WebId                    uniqueidentifier,
    @ItemId                   int,
    @Synchronization          int,
    @Type                     int,
```

[MS-WSSPROG2] — v20120630
Windows SharePoint Services: Content Database Programmability Extensions Communications Version 2 Protocol Specification

Copyright © 2012 Microsoft Corporation.

Release: July 16, 2012
@SequenceNumber int,
@Assembly nvarchar(256),
@Class nvarchar(256),
@Data nvarchar(256),
@Filter nvarchar(256),
@Credential int,
@RequestGuid uniqueidentifier = NULL OUTPUT
);

@DocUrl: The URL in store-relative form of the specified document that has the event receiver.

@Id: The event receiver identifier of the event receiver.

@Name: The name of the event receiver.

@SiteId: The Site Collection identifier of the site collection which contains the document.

@WebId: The Site Identifier (as specified in [MS-WSSFO2], section 2.2.1.11) of the site which contains the document.

@ItemId: This parameter MUST be 0.

@Synchronization: The protocol server MUST ignore this parameter.

@Type: The type of the event receiver. @Type MUST be a value of Event Receiver Type (as specified in [MS-WSSFO2], section 2.2.3.6).

@SequenceNumber: The sequence number (1) of the event receiver. @SequenceNumber MUST be greater than or equal to 0 and less than OR equal to 65535.

@Assembly: The assembly name of the implementation of the event receiver.

@Class: The fully qualified class name of the implementation of the event receiver.

@Data: Additional data persisted on behalf of the event receiver implementation to be passed to the event receiver.

@Filter: Reserved. @Filter MUST be NULL.

@Credential: Reserved. @Credential MUST be 0.

@RequestGuid: The optional request identifier for the current request.

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 event receiver identified by @Id was deleted from the site collection identified by @SiteId.</td>
</tr>
<tr>
<td>3</td>
<td>The document identified by @DocUrl was not found in the site identified by @WebId in the site collection identified by @SiteId.</td>
</tr>
<tr>
<td>87</td>
<td>The deletion failed.</td>
</tr>
</tbody>
</table>

Result Sets: MUST NOT return any result sets.
3.1.4.21 proc_DeleteEventReceiver

The proc_DeleteEventReceiver stored procedure is called to delete the registration of a specified event receiver. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_DeleteEventReceiver(
    @Id uniqueidentifier,
    @Name nvarchar(256),
    @SiteId uniqueidentifier,
    @WebId uniqueidentifier,
    @HostId uniqueidentifier,
    @HostType int,
    @ItemId int,
    @DirName nvarchar(256),
    @LeafName nvarchar(128),
    @Type int,
    @SequenceNumber int,
    @Assembly nvarchar(256),
    @Class nvarchar(256),
    @Data nvarchar(256),
    @Filter nvarchar(256),
    @SourceId varbinary(512),
    @SourceType int,
    @Credential int,
    @ContextType uniqueidentifier,
    @ContextEventType uniqueidentifier,
    @ContextId uniqueidentifier,
    @ContextObjectId uniqueidentifier,
    @ContextCollectionId uniqueidentifier,
    @RequestGuid uniqueidentifier = NULL OUTPUT
);
```

@Id: The Event Receiver identifier (as specified in [MS-WSSFO2], section 2.2.1.3) of the event receiver.

@Name: The name of the event receiver.

@SiteId: The Site Collection identifier of the Site Collection which contains the event host.

@WebId: The Site identifier (as specified in [MS-WSSFO2], section 2.2.1.10) of the Site which contains the event host.

@HostId: The event host identifier of the event host of the event receiver.

@HostType: The type of the event host of the Event receiver. The value MUST be one of Event Host type (as specified in [MS-WSSFO2], section 2.2.3.5).

@ItemId: Reserved. @ItemId MUST be zero.

@DirName: Reserved. @DirName MUST be NULL.

@LeafName: Reserved. @LeafName MUST be NULL.

@Type: The type of the event receiver. @Type MUST be one of Event Receiver Type (as specified in [MS-WSSFO2], section 2.2.3.6).

@SequenceNumber: The sequence number (1) of the event receiver. @SequenceNumber MUST be greater than or equal to zero and less than or equal to 65535.
@Assembly: The assembly name of the implementation of the event receiver.

@Class: The fully qualified class name of the implementation of the event receiver.

@Data: Additional data persisted on behalf of the event receiver implementation to be passed to the event receiver.

@Filter: Reserved. @Filter MUST be NULL.

@SourceId: The event receiver source identifier of the event receiver. This is the Feature identifier (as specified in [MS-WSSFO2], section 2.2.1.4) of the feature if the event receiver is added via a feature. This is the Content type identifier (as specified in [MS-WSSFO2], section 2.2.1.1) of the content type if the event receiver is added via a content type. Otherwise, the event receiver source identifier MUST be NULL.

@SourceType: The Event Receiver Source type of the event receiver. @SourceType MUST be one of the Event Receiver Source Type values.

@Credential: Reserved. @Credential MUST be zero.

@ContextType: The context type identifier of the event receiver.

@ContextEventType: Reserved. @ContextEventType MUST be NULL.

@ContextId: The context identifier of the event receiver.

@ContextObjectId: The context object identifier for the Event Host of the event receiver.

@ContextCollectionId: The context collection identifier of the event receiver.

@RequestGuid: The optional request identifier for the current request.

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 event receiver whose identifier is @Id and Event Receiver type is not identified by 32767 (as specified in [MS-WSSFO2], section 2.2.3.6) was successfully deleted from the site collection identified by @SiteId. All workflow event receivers associated with the workflow context identified by @ContextObjectId that are not used for active workflow were also deleted.</td>
</tr>
<tr>
<td>87</td>
<td>Delete failed because an event receiver was not found whose identifier is @Id or Event Receiver type is not identified by 32767 (as specified in [MS-WSSFO2], section 2.2.3.6) in the site collection identified by @SiteId.</td>
</tr>
</tbody>
</table>

Result Sets: MUST NOT return any result sets.

3.1.4.22 proc_DeleteEventReceiversBySourceId

The proc_DeleteEventReceiversBySourceId stored procedure is called to delete the event receivers registered for a specified event host via a feature or content type. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_DeleteEventReceiversBySourceId(
    @SourceId                varbinary(512),
    @SourceType              int,
    @SiteId                  uniqueidentifier,
```
@SourceId: The event receiver source identifier of the event receiver. This is the Feature Identifier (as specified in [MS-WSSFO2], section 2.2.1.4) of the feature if the event receiver is added via a feature. This is the Content Type Identifier (as specified in [MS-WSSFO2], section 2.2.1.1) of the content type if the event receiver is added via a content type. Otherwise, the event receiver source identifier MUST be NULL.

@SourceType: The Event Receiver Source type of the event receivers to delete. @SourceType MUST be one of the Event Receiver Source Type values.

@SiteId: The Site Collection identifier of the site collection which contains the event host.

@WebId: The Site Identifier (as specified in [MS-WSSFO2], section 2.2.1.11) of the site which contains the event host.

@HostId: The event host identifier of the event host which the event receivers are associated with.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.1.4.23 proc_DeleteInProgressWorkItems

The proc_DeleteInProgressWorkItems stored procedure is called to delete a set of Work Items that meet the criteria specified by the input parameter values. The T-SQL syntax for the stored procedure is as follows:

PROCEDURE proc_DeleteInProgressWorkItems(
    @ProcessingId uniqueidentifier,
    @SiteId uniqueidentifier,
    @ParentId uniqueidentifier,
    @WorkItemType uniqueidentifier,
    @BatchId uniqueidentifier,
    @RequestGuid uniqueidentifier = NULL OUTPUT
);
@WorkItemType: The Work Item type identifier of the Work Item type. The server MUST only delete Work Items associated with this Work Item type. MUST NOT be NULL.

@BatchId: The Work Item Batch identifier of the Work Item Batch. If the parameter is not NULL, then the server MUST only delete Work Items associated with this Work Item Batch. If the parameter is NULL, then the server MUST delete Work Items that meet the criteria specified by the other parameters regardless of associated Work Item Batch.

@RequestGuid: The optional request identifier for the current request.

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

Result Sets: MUST NOT return any result sets.

3.1.4.24 proc_DeleteSmartPagePersonalization

The proc_DeleteSmartPagePersonalization stored procedure is called to delete personalizations from all Web Parts on the Web Part Page and to delete all personal Web Parts from the Web Part Page. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_DeleteSmartPagePersonalization(
  @SiteId                   uniqueidentifier,
  @DirName                  nvarchar(256),
  @LeafName                 nvarchar(128),
  @UserId                   int,
  @RequestGuid              uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The Site Collection identifier for the Site Collection which contains the Page specified by @DirName and @LeafName. MUST NOT be NULL.

@DirName: The Directory Name of the Web Part Page from which to delete personalizations and personal Web Parts. MUST NOT be NULL.

@LeafName: The Leaf Name of the Web Part Page from which to delete personalizations and personal Web Parts. MUST NOT be NULL.

@UserId: The User Identifier (as specified in [MS-WSSFO2], section 2.2.1.13) for which to delete personalizations and personal Web Parts. If @UserId is NULL, the stored procedure MUST delete personalizations and personal Web Parts for every user. If @UserId is not NULL, the stored procedure MUST ONLY delete personalizations and personal Web Parts for the user specified by @UserId.

@RequestGuid: The optional request identifier for the current request.

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 completion.</td>
</tr>
<tr>
<td>1</td>
<td>Internal SQL error.</td>
</tr>
<tr>
<td>2</td>
<td>The Page specified by @DirName and @LeafName does not exist or has been deleted.</td>
</tr>
</tbody>
</table>

Result Sets: MUST NOT return any result sets.
3.1.4.25 proc_DeleteWebPart

The proc_DeleteWebPart stored procedure is called to delete Web Part from the Web Part Page. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_DeleteWebPart (
    @SiteId                   uniqueidentifier,  
    @DirName                  nvarchar(256),    
    @LeafName                 nvarchar(128),    
    @Level                    tinyint OUTPUT,    
    @UserId                   int,              
    @WebPartID                uniqueidentifier, 
    @RequestGuid              uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The Site Collection identifier for the Site Collection which contains the Web Part specified by @WebPartID. MUST NOT be NULL.

@DirName: The Directory Name of the Web Part Page that contains the Web Part specified by @WebPartID. MUST NOT be NULL.

@LeafName: The Leaf Name of the Web Part Page that contains the Web Part specified by @WebPartID. MUST NOT be NULL.

@Level: This is an input/output parameter. On input, this is the Publishing Level value of the Page specified by @LeafName that contains the Web Part specified by @WebPartID. On output, this is the Publishing Level of the Page specified by @LeafName after the Web Part specified by @WebPartID is deleted. MUST NOT be NULL.

@UserId: The User Identifier (as specified in [MS-WSSFO2], section 2.2.1.13) of the user which is deleting the Web Part specified by @WebPartID.

@WebPartID: The Web Part Identifier (as specified in [MS-WSSFO2], section 2.2.1.15) of the Web Part to be deleted. MUST NOT be NULL.

@RequestGuid: The optional request identifier for the current request.

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 completion.</td>
</tr>
<tr>
<td>1</td>
<td>Internal SQL error.</td>
</tr>
<tr>
<td>2</td>
<td>The Page specified by @DirName, @LeafName, and @Level does not exist or has been deleted.</td>
</tr>
<tr>
<td>3</td>
<td>The Web Part Page is Moderated or has minor version control enabled, and a new version of the Web Part Page cannot be created because a unique name for it cannot be created.</td>
</tr>
<tr>
<td>5</td>
<td>The Web Part specified by @WebPartID lives in a Page different from the Page specified by @DirName and @LeafName.</td>
</tr>
<tr>
<td>12</td>
<td>Cannot delete a personalized Web Part from a Page that is Checked Out.</td>
</tr>
<tr>
<td>33</td>
<td>The Page specified by @DirName, @LeafName, and @Level is not the Current Version.</td>
</tr>
<tr>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>87</td>
<td>The Page specified by @DirName, @LeafName, and @Level does not exist or has been deleted.</td>
</tr>
<tr>
<td>158</td>
<td>The Page specified by @DirName and @LeafName needs to be Checked Out because the Page lives in a Document Library with Required Checkout set.</td>
</tr>
<tr>
<td>160</td>
<td>Need to create a new version of the Page specified by @DirName and @LeafName, but no user is specified by @UserId.</td>
</tr>
<tr>
<td>212</td>
<td>Need to create a new version of the Page specified by @DirName and @LeafName, but the Site Collection specified by @SiteId is locked.</td>
</tr>
<tr>
<td>1816</td>
<td>Need to create new version of Page specified by @DirName and @LeafName, but the Site Collection specified by @SiteId has exceeded its Quota.</td>
</tr>
</tbody>
</table>

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

### 3.1.4.26 proc_DeleteWebPartPersonalization

The proc_DeleteWebPartPersonalization stored procedure is called to delete personalizations from the specified Web Part. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_DeleteWebPartPersonalization(
    @SiteId                   uniqueidentifier,
    @DirName                  nvarchar(256),
    @LeafName                 nvarchar(128),
    @UserId                   int,
    @WebPartID                uniqueidentifier,
    @RequestGuid              uniqueidentifier = NULL OUTPUT
);
```

- **@SiteId:** The Site Collection identifier for the Site Collection which contains the Web Part specified by @WebPartID. MUST NOT be NULL.
- **@DirName:** The Directory Name of the Web Part Page that contains the Web Part specified by @WebPartID. MUST NOT be NULL.
- **@LeafName:** The Leaf Name of the Web Part Page that contains the Web Part specified by @WebPartID. MUST NOT be NULL.
- **@UserId:** The User Identifier (as specified in [MS-WSSFO2], section 2.2.1.13) of the user for which to delete personalizations. If @UserId is NULL, the stored procedure MUST delete personalizations from the Web Part specified in @WebPartID for every user. If @UserId is not NULL, the stored procedure MUST delete personalizations from the Web Part specified in @WebPartID for the user specified by @UserId.
- **@WebPartID:** The Web Part Identifier (as specified in [MS-WSSFO2], section 2.2.1.15) of the Web Part from which to delete personalizations. MUST NOT be NULL.
- **@RequestGuid:** The optional request identifier for the current request.

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

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful completion.</td>
</tr>
<tr>
<td>1</td>
<td>Internal SQL error.</td>
</tr>
<tr>
<td>2</td>
<td>The Page specified by @DirName and @LeafName does not exist or has been deleted.</td>
</tr>
<tr>
<td>5</td>
<td>The Web Part specified by @WebPartID lives in a Page different from the Page specified by @DirName and @LeafName.</td>
</tr>
</tbody>
</table>

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

### 3.1.4.27 proc_DeleteWebPartWhileSaving

The `proc_DeleteWebPartWhileSaving` stored procedure is called to delete a Web Part from the Shared View of the Web Part Page. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_DeleteWebPartWhileSaving(
    @SiteId                  uniqueidentifier,
    @DirName                 nvarchar(256),
    @LeafName                nvarchar(128),
    @Level                   tinyint,
    @WebPartID               uniqueidentifier,
    @RequestGuid             uniqueidentifier = NULL OUTPUT
);
```

- **@SiteId**: The Site Collection identifier of the Site Collection which contains the Web Part Page.
- **@DirName**: The Directory Name of the Web Part Page.
- **@LeafName**: The Leaf Name of the Web Part Page.
- **@Level**: The Publishing Level of the Web Part Page.
- **@WebPartID**: The Web Part Identifier (as specified in [MS-WSSPROG2], section 2.2.1.15) of the Web Part to be deleted from the Web Part Page.
- **@RequestGuid**: The optional request identifier for the current request.

**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 completion.</td>
</tr>
<tr>
<td>2</td>
<td>The Web Part Page cannot be found or @SiteId, @DirName or @LeafName is NULL.</td>
</tr>
<tr>
<td>33</td>
<td>The Web Part Page is not the Current Version.</td>
</tr>
<tr>
<td>5</td>
<td>The Web Part is not in a Shared View.</td>
</tr>
<tr>
<td>1</td>
<td>An internal SQL error occurred.</td>
</tr>
</tbody>
</table>

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

The proc_DeleteZoneWebPartsWhileSaving stored procedure is called to delete all the Web Parts in a Web Part Zone from the Shared View of the Web Part Page. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_DeleteZoneWebPartsWhileSaving(
    @SiteId                   uniqueidentifier,
    @DirName                  nvarchar(256),
    @LeafName                 nvarchar(128),
    @PageUrlID                uniqueidentifier,
    @Level                    tinyint,
    @WebPartZoneID            nvarchar(64),
    @RequestGuid              uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The Site Collection identifier of the Site Collection which contains the Web Part Page. MUST NOT be NULL.

@DirName: The Directory Name of the Web Part Page. MUST NOT be NULL.

@LeafName: The Leaf Name of the Web Part Page. MUST NOT be NULL.

@PageUrlID: The Document Identifier (as specified in [MS-WSSFO2], section 2.2.1.2) of the Web Part Page.

@Level: The Publishing Level of the Web Part Page.

@WebPartZoneID: The Web Part Zone identifier of the Web Part Zone of the Web Part Page. MUST NOT be NULL.

@RequestGuid: The optional request identifier for the current request.

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 completion.</td>
</tr>
<tr>
<td>33</td>
<td>The Web Part Page is not the Current Version.</td>
</tr>
<tr>
<td>1</td>
<td>An internal SQL error occurred.</td>
</tr>
</tbody>
</table>

Result Sets: MUST NOT return any result sets.

3.1.4.29 proc_DisableAssociationsForTemplate

The proc_DisableAssociationsForTemplate stored procedure is called to disable Workflow associations based on a workflow template. When a Workflow association is disabled, no new Workflows can be created from that Workflow association. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_DisableAssociationsForTemplate (
    @SiteId          uniqueidentifier,
    @BaseId          uniqueidentifier,
    @RequestGuid     uniqueidentifier = NULL OUTPUT
);  
```
@SiteId: The Site Collection identifier of the Site Collection.

@BaseId: The Workflow Template identifier of the workflow template. The server MUST disable all Workflow associations in the Site Collection based on the workflow template. The server MUST NOT allow any Workflow associations in the Site Collection based on the workflow template to create any new Workflows.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.1.4.30 proc_DropWorkflow

The proc_DropWorkflow stored procedure is called to delete a workflow. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE dbo.proc_DropWorkflow(
    @WorkflowInstanceId uniqueidentifier,
    @SiteId uniqueidentifier,
    @WebId uniqueidentifier,
    @ListId uniqueidentifier,
    @RequestGuid uniqueidentifier = NULL OUTPUT)
```

@WorkflowInstanceId: The Workflow identifier of the Workflow. The server MUST delete the Workflow.

@SiteId: The Site Collection identifier of the Site Collection which contains the Workflow. The server MUST update the site collection quota (3.1.1.5) to remove the space used by the deleted workflow.

@WebId: The protocol server MUST ignore this parameter.

@ListId: The protocol server MUST ignore this parameter.

@RequestGuid: The optional request identifier for the current request.

Result Sets: MUST NOT return any result sets.

3.1.4.31 proc_DropWorkflowAssociation

The proc_DropWorkflowAssociation stored procedure is called to delete a workflow association and its associated workflows. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE dbo.proc_DropWorkflowAssociation(
    @SiteId uniqueidentifier,
    @Id uniqueidentifier,
    @DropAll int = 0,
    @RequestGuid uniqueidentifier = NULL OUTPUT)
```
@SiteId: The Site Collection identifier of the Site Collection which contains the Workflow association. The server MUST update the site collection quota (section 3.1.1.5) to remove the spaced used by the deleted workflows.

@Id: The Workflow association identifier of the Workflow association. The server MUST delete the Workflow association and all Workflows based on the Workflow association.

@DropAll: This parameter specifies whether the server throttles the deletion process. This value MUST be 0 or 1. When set to 1, the server MUST perform the entire deletion process immediately. When set to 0, the server MUST throttle the deletion process by deleting an limited, implementation defined number of Workflows immediately and, if the limit is reached, by marking the remaining Workflows and the Workflow association for deferred deletion by proc_AutoCleanupWorkflows.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.1.4.32 proc_DropWorkItem

The proc_DropWorkItem stored procedure is called to delete an existing Work Item from the set of pending Work Items for a Site Collection. The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_DropWorkItem(
    @SiteId uniqueidentifier,
    @Id uniqueidentifier,
    @RequestGuid uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The Site Collection identifier of the Site Collection of the Work Item. MUST NOT be NULL.

@Id: The Work Item identifier. The server MUST only delete the Work Item associated with this Work Item identifier. MUST NOT be NULL.

@RequestGuid: The optional request identifier for the current request.

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

Result Sets: MUST NOT return any result sets.

3.1.4.33 proc_EnableDeclarativeWorkflowAssociations

The proc_EnableDeclarativeWorkflowAssociations stored procedure is called to enable or disable all Declarative Workflow associations contained in a Site Collection. When a Workflow association is disabled, no new Workflows can be created based on that Workflow association. The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE dbo.proc_EnableDeclarativeWorkflowAssociations(
    @SiteId uniqueidentifier,
    @Enabled int,
    @RequestGuid uniqueidentifier = NULL OUTPUT
);
```
@SiteId: The Site Collection identifier of the Site Collection which contains the Workflow associations.

@Enabled: This parameter determines whether the Workflow associations are enabled or disabled. This value MUST be 0 or 1. When set to 1, the server MUST enable all Declarative Workflow associations in the Site Collection. When set to 0, the server MUST disable all the Declarative Workflow associations in the Site Collection, and MUST NOT allow any new Workflows to be created from the disabled Workflow associations.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.1.4.34 proc_EnumerateWebPartsForList

The proc_EnumerateWebPartsForList stored procedure is called to return properties of Web Parts in shared views from Published Web Part Pages contained within the specified list. The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_EnumerateWebPartsForList(
    @SiteId                   uniqueidentifier,
    @WebId                    uniqueidentifier,
    @ListId                   uniqueidentifier,
    @RequestGuid              uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The Site Collection identifier for the Site Collection which contains the list.

@WebId: The Site Identifier (as specified in [MS-WSSFO2], section 2.2.1.11) of the Site which contains the list.

@ListId: The List Identifier (as specified in [MS-WSSFO2], section 2.2.1.5) of the list.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST return the following result set:

3.1.4.34.1 Web Parts Result Set

The Result Set is defined in section 2.2.6.4.

3.1.4.35 proc_EnumerateWebPartsForWeb

The proc_EnumerateWebPartsForWeb stored procedure is called to return properties of Web Parts in Shared Views of the specified Site. Only Web Parts from Published Web Part Pages are returned. The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_EnumerateWebPartsForWeb(
```

[MWSSPROG2] — v20120630
Windows SharePoint Services: Content Database Programmability Extensions Communications Version 2 Protocol Specification

Copyright © 2012 Microsoft Corporation.

Release: July 16, 2012
`@SiteId` uniqueidentifier,
`@WebId` uniqueidentifier,
`@RequestGuid` uniqueidentifier = NULL OUTPUT
}

**@SiteId**: The Site Collection identifier of the site collection which contains the Site.

**@WebId**: The Site Identifier, as specified in [MS-WSSFO2] section 2.2.1.11, of the Site.

**@RequestGuid**: The optional request identifier for the current request.

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

**Result Sets**: MUST return the following result set:

### 3.1.4.35.1 Web Parts Result Set

Web Parts Result Set returns properties of the Web Parts. There MUST be one row per Web Part in this Result Set. The T-SQL syntax for the result set is as follows:

```
.tp_ID uniqueidentifier,
.tp_ListId uniqueidentifier,
.tp_Type tinyint,
.tp_Flags int,
.tp_DisplayName nvarchar(255),
.tp_Version int,
{DocumentUrl} nvarchar(385),
.tp_PartOr\dder int,
.tp_ZoneID nvarchar(64),
.tp_IsIncluded bit,
.tp_FrameState tinyint,
.tp_WebPartTypeId uniqueidentifier,
.tp_Assembly nvarchar(255),
.tp_Class nvarchar(255),
.tp_SolutionId uniqueidentifier,
.tp_AllUsersProperties varbinary(max),
.tp_PerUserProperties varbinary(max),
.tp_WebPartIdProperty nvarchar(255),
.tp_Cache varbinary(max),
.tp_Source nvarchar(max),
.tp_View nvarchar(max)
```

**tp_ID**: The Web Part Identifier, as specified in [MS-WSSFO2], section 2.2.1.15. This value MUST not be NULL.

**tp_ListId**: The List Identifier, as specified in [MS-WSSFO2], section 2.2.1.5, of the List to which the Web Part refers.

**tp_Type**: The Page Type, as specified in [MS-WSSFO2], section 2.2.3.14, of the Web Part Page that contains the Web Part.

**tp_Flags**: The View Flags, as specified in [MS-WSSFO2], section 2.2.2.12, of the Web Part.

**tp_DisplayName**: The display name of the Web Part.

**tp_Version**: This value MUST be ignored.
{DocumentUrl}: The store-relative form URL of the Web Part Page that contains the Web Part. This value MUST NOT be NULL.

Tp_PartOrder: The Web Part zone index of the Web Part.

Tp_ZoneID: The Web Part zone identifier of the Web Part.

Tp_IsIncluded: 1 if the Web Part is included the Web Part Page; 0 if the Web Part is not included. This value MUST NOT be NULL.

Tp_FrameState: The Web Part chrome state of the Web Part. This value MUST NOT be NULL.

Tp_WebPartTypeId: The Web Part type identifier of the Web Part.

Tp_Assembly: The fully qualified name of the assembly that implements the web part.

Tp_Class: The name of the .NET class that implements the Web Part.

Tp_SolutionId: The identifier of the sandboxed solution or site solution that installed the Web Part.

Tp_AllUsersProperties: A binary payload containing 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.

Tp_PerUserProperties: A binary payload containing zero or more personalizable properties on the Web Part. If this value is NULL, then default values will be used for all of the personalizable properties on the Web Part.

Tp_WebPartIdProperty: The HTML (HyperText Markup Language) ID attribute of the Web Part. May be NULL. If not NULL, it MUST be unique per Web Part Page.

Tp_Cache: Private data cache of the Web Part.

Tp_Source: The Web Part properties of the Web Part in WPV2:WebPart format (as specified in [MS-WPPS], section 2.2.3.2), WPV3:WebPart format (as specified in [MS-WPPS], section 2.2.3.3) or WebParts format (as specified in [MS-WPPS], section 2.2.3.1). The protocol client can determine which format is used by comparing the value against the schemas for the formats. The value will be NULL if the properties are compressed and stored in Tp_AllUserProperties and Tp_PerUserProperties.

Tp_View: The CAML of the Web Part.

3.1.4.36 proc_EnumResourceWarningSites

The proc_EnumResourceWarningSites stored procedure is called to return information about all of the site collections in a content database that have exceeded the warning level for their resource quota. The T-SQL syntax for the stored procedure is as follows:

PROCEDURE proc_EnumResourceWarningSites ();

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

Result Sets: This procedure MUST return the Resource Warning Site Collections Result Set.
3.1.4.36.1 Resource Warning Site Collections Result Set

The Resource Warning Site Collections Result Set returns the list of all site collections that have exceeded the warning level for their resource quota. The T-SQL syntax for the result set is as follows:

```sql
Id uniqueidentifier NOT NULL,
BitFlags int NOT NULL,
CurrentResourceUsage float NOT NULL,
AverageResourceUsage float NOT NULL,
ResourceUsageWarning float NOT NULL,
ResourceUsageMaximum float NOT NULL;
```

**Id:** The site collection identifier of the site collection that has exceeded the warning level for its resource quota.

**BitFlags:** The Site Collection Flags value, as defined in [MS-WSSFO2], section 2.2.2.9, describing the configuration of the site collection.

**CurrentResourceUsage:** The resource usage value for the site collection during the current monitoring interval for resource usage.

**AverageResourceUsage:** The mean resource usage value for the site collection over the available number of monitoring intervals for resource usage.

**ResourceUsageWarning:** The warning level for a resource quota for this site collection.

**ResourceUsageMaximum:** The maximum level for a resource quota for this site collection.

3.1.4.37 proc_FailOverInProgressWorkItems

The proc_FailOverInProgressWorkItems stored procedure is called to mark a set of Work Items as not In Progress Work Items. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_FailOverInProgressWorkItems(
    @ProcessingId uniqueidentifier,
    @RequestGuid uniqueidentifier = NULL OUTPUT
);
```

**@ProcessingId:** The Work Item Processing identifier of the Work Item Process. This parameter MUST NOT be NULL. For each Work Item associated with the given Work Item Processing identifier for which the Work Item Delivery Date has passed, the server MUST do the following:

- Mark the Work Item as not In Progress Work Items.
- Set the Work Item Processing identifier of the Work Item to NULL.

**@RequestGuid:** The optional request identifier for the current request.

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

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

The proc_GetAllResourceUsageForSiteToday stored procedure is called to return resource usage values for a given site collection during the current monitoring interval for resource usage. The T-SQL syntax for the stored procedure is as follows:

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

@SiteId: The site collection identifier for the site collection for which resource usage values are requested.

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

**Result Sets:** This procedure MUST return the Site Collection Daily Resource Usage Result Set.

3.1.4.38.1 Site Collection Daily Resource Usage Result Set

The Site Collection Daily Resource Usage Result Set returns resource usage values for each monitored resource measure reported for the specified site collection over the current monitoring interval for resource usage. The T-SQL syntax for the result set is as follows:

```sql
ResourceId uniqueidentifier NOT NULL,  
TotalResourceUsage float NOT NULL;
```

ResourceId: The identifier of the monitored resource measure for this resource usage value.

TotalResourceUsage: The resource usage value for the monitored resource measure for the specified site collection over the current monitoring interval for resource usage.

3.1.4.39 proc_GetAllWebPartsOnPage

The proc_GetAllWebPartsOnPage stored procedure is called to return information about all of the Web Parts on a Web Part Page. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_GetAllWebPartsOnPage(  
    @SiteId                  uniqueidentifier,  
    @CurrentWebId            uniqueidentifier,  
    @AllUsers                bit,  
    @SystemID                varbinary(512),  
    @DirName                 nvarchar(256),  
    @LeafName                nvarchar(128),  
    @Level                   tinyint,  
    @PrefetchListScope       bit,  
    @Threshold.RowCount       int,  
    @GetViewBodies           bit,  
    @RequestGuid             uniqueidentifier = NULL OUTPUT  
) ;
```

@SiteId: The Site Collection identifier of the site collection which contains the Web Part Page from which to get Web Parts.
@CurrentWebId: The identifier of the Site containing the Web Part Page from which to get Web Parts.

@AllUsers: Specifies whether to return Web Parts for the Shared View or personal View of the Web Part Page. If set to 1, Web Parts for the Shared View are returned in the [MS-WSSFO2].proc_FetchDocForHttpGet.Web Parts Metadata, Non-Personalized Result Set (as specified in [MS-WSSFO2] section 3.1.5.19.18). If set to 0, Web Parts personalized for the current user are returned in the [MS-WSSFO2].proc_FetchDocForHttpGet.WebParts Metadata, Personalized Result Set (as specified in [MS-WSSFO2], section 3.1.5.19.17).

@SystemID: The SystemID of the user originating the request or NULL to indicate an anonymous user if @AllUsers is 0.

@DirName: The Directory Name of the Web Part Page.

@LeafName: The Leaf Name of the Web Part Page.

@Level: The Publishing Level of the Web Part Page from which to get Web Parts.

@PrefetchListScope: This value MUST be set to 1.

@ThresholdRowCount: The maximum number of rows to return in the List Metadata Result Set.

@GetViewBodies: This value MUST be set to 1.

@RequestGuid: The optional request identifier for the current request.

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 completion.</td>
</tr>
<tr>
<td>-2147467259</td>
<td>An error occurred while the stored procedure was running.</td>
</tr>
<tr>
<td>1</td>
<td>An error occurred while the stored procedure was running.</td>
</tr>
<tr>
<td>2</td>
<td>Specified Web Part Page does not exist.</td>
</tr>
</tbody>
</table>

Result Sets: MUST return zero, two or four result sets. No result set is returned when this stored procedure does not complete successfully. If two or four result sets are returned, first, either the Web Parts Metadata, Non-Personalized or the Web Parts Metadata, Personalized Result Set MUST be returned. Second, the list Metadata Result Set MUST be returned. If the list Metadata Result Set is empty then the rest of the result sets MUST NOT be returned.

3.1.4.39.1 Web Parts Metadata, Non-Personalized Result Set

If @AllUsers is 1, [MS-WSSFO2].proc_FetchDocForHttpGet.Web Parts Metadata, Non-Personalized Result Set MUST be returned (as specified in [MS-WSSFO2] section 3.1.5.19.18).

3.1.4.39.2 Web Parts Metadata, Personalized Result Set

If @AllUsers is 0, [MS-WSSFO2].proc_FetchDocForHttpGet.Web Parts Metadata, Personalized Result Set MUST be returned (as specified in [MS-WSSFO2], section 3.1.5.19.17).
3.1.4.39.3 List Metadata, Result Set

[MS-WSSFO2].proc_FetchDocForHttpGet.List Metadata, Result Set MUST be returned (as specified in [MS-WSSFO2] section 3.1.5.19.19).

3.1.4.39.4 List Event Receivers, Result Set

If List Metadata, Result Set is NOT empty then [MS-WSSFO2].proc_FetchDocForHttpGet.List Event Receivers, Result Set MUST be returned (as specified in [MS-WSSFO2] section 3.1.5.19.20).

3.1.4.39.5 List Security Information, Result Set

If List Metadata, Result Set is NOT empty then [MS-WSSFO2].proc_FetchDocForHttpGet.List Security Information, Result Set MUST be returned (as specified in [MS-WSSFO2] section 3.1.5.19.21).

3.1.4.40 proc_GetAverageDailyResourceUsageForSite

The proc_GetAverageDailyResourceUsageForSite stored procedure is called to return the mean resource usage value of each monitored resource measure for a given site collection. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_GetAverageDailyResourceUsageForSite (  
    @SiteId                  uniqueidentifier,  
    @RetentionDays           int
);  

@SiteId: The site collection identifier of the site collection for which to retrieve mean resource usage values.

@RetentionDays: The number of monitoring intervals for which resource usage values are retained.

Return Code values: An integer value which MUST be 0.

Result Sets: This procedure MUST return the Site Collection Average Daily Resource Usage Result Set.

3.1.4.40.1 Site Collection Average Daily Resource Usage Result Set

The Site Collection Average Daily Resource Usage Result Set returns the mean resource usage value for each monitored resource measure for the specified site collection. The T-SQL syntax for the result set is as follows:

```sql
ResourceId            uniqueidentifier NOT NULL,  
AvgResourceUsage      float NOT NULL;
```

ResourceId: The identifier of a monitored resource measure.

AvgResourceUsage: The mean resource usage value of the monitored resource measure for the specified site collection over the number of monitoring intervals specified by @RetentionDays.
3.1.4.41 proc_GetContextCollectionEventReceivers

The proc_GetContextCollectionEventReceivers stored procedure is called to retrieve, for a specific Site Collection and Context Collection, a collection of event receivers of a specific Context type. The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_GetContextCollectionEventReceivers(
    @SiteId                   uniqueidentifier,
    @ContextCollectionId      uniqueidentifier,
    @ContextType              uniqueidentifier = NULL,
    @RequestGUid              uniqueidentifier = NULL
) OUTPUT
```

@SiteId: The Site Collection identifier of the site collection for which the event receivers are requested to be returned. This value MUST NOT be NULL.

@ContextCollectionId: The context collection identifier for the Context Collection for which the event receivers are requested to be returned.

@ContextType: The context type identifier for the Context type the event receivers have to match. The default is NULL. When this value is NULL, event receivers of any Context type are returned.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST return the following result set:

3.1.4.41.1 Event Receivers Result Set

The Result Set is defined in the Event Receivers Result Set (as specified in [MS-WSSFO2], section 2.2.5.9). The Result Set MUST also include an additional NULL column at the end of the result set.

3.1.4.42 proc_GetContextObjectEventReceivers

The proc_GetContextObjectEventReceivers stored procedure is called to retrieve a list of Event Receivers and optionally remove an Event Receiver. The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_GetContextObjectEventReceivers(
    @SiteId                  uniqueidentifier,
    @WebId                   uniqueidentifier,
    @HostId                  uniqueidentifier,
    @ContextObjectId         uniqueidentifier,
    @ContextObjectItemId     int,
    @DeleteHostLookupId      uniqueidentifier = NULL,
    @HostType                int = NULL,
    @RequestGuid             uniqueidentifier = NULL OUTPUT
) OUTPUT
```

@SiteId: The Site Collection identifier of the site collection which contains the Site for which the Event Receivers are be requested. This value MUST NOT be NULL.

@WebId: The protocol server MUST ignore this value.
@HostId: The event host identifier of the event host for which the event receivers are requested.

@ContextObjectId: The context object identifier for the Context Object of the Workflow receiver process associated with the Event Receivers to be requested. If this value is not NULL, the protocol server MUST ignore the values of @HostId and @ContextObjectItemId.

@ContextObjectItemId: The context object identifier of the Event Host for which the registered event receivers are requested.

@DeleteHostLookupId: The Event Receiver Identifier (as specified in [MS-WSSFO2], section 2.2.1.3) for the Event Receiver to be optionally removed. If the value is NULL, no Event Receivers will be deleted. If the value is NOT NULL, the Event Receiver, with Site Collection identifier property equal to @SiteId and Event Receiver Identifier (as specified in [MS-WSSFO2], section 2.2.1.3) equal to @DeleteHostLookupId, will be deleted. The default is NULL.

@HostType: The Event Host Type (as specified in [MS-WSSFO2], section 2.2.3.5) of the Event Receivers that are requested. If this parameter is NOT NULL, the results are filtered for HostType=@HostType. If this parameter is NULL, no result filtering is performed and Event Receivers with any value of Event Host Type (as specified in [MS-WSSFO2], section 2.2.3.5) are returned. The default is NULL.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST return the following result set when (a) @DeleteHostLookupId is set to NULL or (b) DeleteHostLookupId is NOT NULL and the Event Receiver it refers to exists and its Event Receiver type is equal to 32767 (as specified in [MS-WSSFO2], section 2.2.3.6). In the case where @DeleteHostLookupId is NOT NULL and the Event Receiver it refers to does not exist OR its Event Receiver type is not equal to 32767 (as specified in [MS-WSSFO2], section 2.2.3.6), the stored procedure MUST NOT return a result set.

3.1.4.42.1 Event Receivers Result Set

This Result Set will be filtered by the Site through @SiteId and the context object identifier through one of either @ContextObjectId or the combination of @HostId and @ContextObjectItemId. The protocol client MUST specify both the site and the context object. The optional parameter @HostType can be used to further filter the results. Note that when @HostType is set to NULL, no HostType filtering is performed and all rows with any value of HostType are returned. The Result Set is defined in the Event Receivers Result Set (as specified in [MS-WSSFO2], section 2.2.5.9), plus one additional NULL column appended to the end of the result set.

3.1.4.43 proc_GetDocEventReceivers

The proc_GetDocEventReceivers stored procedure is called to read all event receivers registered for a specified document. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_GetDocEventReceivers(
    @DocSiteId uniqueidentifier,
    @DocWebId uniqueidentifier,
    @DocUrl nvarchar(260),
    @RequestGuid uniqueidentifier = NULL OUTPUT
);
```

@DocSiteId: The site collection identifier of the site collection which contains the 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 completion.</td>
</tr>
<tr>
<td>3</td>
<td>The document does not exist.</td>
</tr>
</tbody>
</table>

Result Sets: MUST return the following result set that contains one row for each of the event receivers registered for the specified document when the return code is 0. proc_GetDocEventReceivers MUST NOT return a result set when the return code is not 0.

3.1.4.43.1 Event Receivers Result Set

The Result Set is defined in the Event Receivers Result Set (as specified in [MS-WSSFO2] section 2.2.5.9).

3.1.4.44 proc_GetListItemWorkflows

The proc_GetListItemWorkflows stored procedure is called to obtain a set of Workflows. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE dbo.proc_GetListItemWorkflows(
    @SiteId                   uniqueidentifier,
    @WebId                    uniqueidentifier,
    @ListId                   uniqueidentifier,
    @ItemId                   int,
    @WorkflowInstanceId       uniqueidentifier,
    @TemplateId               uniqueidentifier,
    @InclusiveFilterState     int = 0xFFFFFFFF,
    @ExclusiveFilterState     int = 0,
    @Limit                    int = 0,
    @LimitFlags               int = 0,
    @RequestGuid              uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The Site Collection identifier of the Site Collection which contains the Workflows.

@WebId: The Site identifier (as specified in [MS-WSSFO2], section 2.2.1.11) of the Site which contains the Workflows.

@ListId: The list identifier of the list which contains the list Items the Workflows were created for. If this value is NULL, the server MUST include all lists.

@ItemId: The list Item identifier (as specified in [MS-WSSFO2], section 2.2.1.6) of the list Items the Workflows were created for. If @ListId is NULL, this value MUST be NULL. If this value is NULL, the server MUST include all list Items.
@WorkflowInstanceId: The Workflow identifier of the Workflow. If this value is NULL, the server MUST include all Workflows. If this value is not NULL, the server MUST ignore @WebId, @ListId, @ItemId and @TemplateId and return only one row in the Result Set which contains the Workflow specified by @WorkflowInstanceId.

@TemplateId: The Workflow Template identifier of the workflow template. If this value is NULL, the server MUST include all workflow templates.

@InclusiveFilterState: A workflow internal state bitmask. The server MUST include only Workflows that have at least one internal state flag in common with the bitmask (that is, Workflow.InternalState & @InclusiveFilterState <> 0).

@ExclusiveFilterState: A workflow internal state bitmask. The server MUST exclude all Workflows that have any internal state flags in common with the bitmask (that is, Workflow.InternalState & @ExclusiveFilterState <> 0).

@Limit: The optional limit for the number of Workflows returned in the Result Set. This value MUST be a positive integer or 0. If this value is not 0 the server MUST limit the number of returned Workflows in the Result Set to this value. If this value is 0 the server MUST NOT limit the number of returned Workflows in the Result Set.

@LimitFlags: The protocol server MUST ignore this parameter.

@RequestId: The optional request identifier for the current request.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST return the list Item Workflows Result Set specified in section 2.2.6.1. The InstanceData and ProcessingId columns in the Result Set must be NULL and the InstanceDataSize column MUST contain the value 0.

3.1.4.44.1 List Item Workflows Result Set

The Result Set is defined in section 2.2.6.1.

3.1.4.45 proc_GetListItemWorkflowWithInstanceDataAndLock

The proc_GetListItemWorkflowWithInstanceDataAndLock stored procedure is called to lock a workflow and get back a result set for the workflow. The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_GetListItemWorkflowWithInstanceDataAndLock(
    @SiteId                  uniqueidentifier,
    @WebId                   uniqueidentifier,
    @ListId                  uniqueidentifier,
    @ItemId                  int,
    @WorkflowInstanceId      uniqueidentifier,
    @HasInstanceData         int OUTPUT,
    @RequestGuid             uniqueidentifier = NULL OUTPUT
);  
```

@SiteId: The Site Collection identifier of the Site Collection which contains the Workflow.

@WebId: The Site Identifier (as specified in [MS-WSSFO2], section 2.2.1.11) of the Site which contains the Workflow.
@ListId: The protocol server MUST ignore this parameter.

@ItemId: The protocol server MUST ignore this parameter.

@WorkflowInstanceId: The Workflow identifier of the Workflow. The server MUST attempt to lock the Workflow.

@HasInstanceData: The server MUST ignore the input value of this parameter. If the server locked the Workflow, the server MUST set the output value to 1. If the server did not lock the workflow, the server MUST set the output value to 0.

@RequestGuid: The optional request identifier for the current request.

**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 completion.</td>
</tr>
<tr>
<td>5</td>
<td>Error: Access denied.</td>
</tr>
<tr>
<td>19</td>
<td>Error: The workflow was not found or was locked.</td>
</tr>
<tr>
<td>82</td>
<td>Error: Failed to lock the workflow.</td>
</tr>
</tbody>
</table>

**Result Sets:** MUST return the [List Item Workflows Result Set](#) with exactly one row containing the Workflow specified by @WorkflowInstanceId. If the Workflow was successfully locked, the InstanceData, InstanceDataSize and ProcessingId columns MUST contain the instance data for the Workflow, the instance data size, and the identifier of the computer processing the Workflow, respectively. If the workflow was not successfully locked, the InstanceData and ProcessingId columns MUST be NULL and the InstanceDataSize column MUST contain the value 0.

### 3.1.4.46 proc_GetListWebParts

The proc_GetListWebParts stored procedure is called to return a Result Set of List View Web Parts, List Form Web Parts, Data View Web Parts, and Data Form Web Parts associated with the specified list in Web Part pages. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_GetListWebParts(
    @ListId                   uniqueidentifier,
    @ViewId                   uniqueidentifier,
    @UserID                   int,
    @DocVersion               int,
    @bGetAllLevel             bit,
    @bGetDeleted              bit = 0,
    @bGetAllUsers             bit = 0,
    @SiteId                   uniqueidentifier,
    @RequestGuid              uniqueidentifier = NULL OUTPUT
);
```

@ListId: The list identifier (as specified in [MS-WSSFO2], section 2.2.1.5).

@ViewId: The GUID of the list view, or NULL for the default view.

@UserID: The User Identifier (as specified in [MS-WSSFO2], section 2.2.1.13) of the current user.
@bGetAllLevel: The parameter determines whether to include Web Parts from Web Part Pages with all Publishing Levels. When this parameter is set to 1, the Result Set MUST include Web Parts from Web Part Pages with all Publishing Levels. When set to 0, the Result Set MUST only include Web Parts from Web Part Pages with the highest Publishing Level that the current user has permission to view.

@DocVersion: The document version of the web part page containing the view of the list, or 0 for the current version. MUST NOT be NULL.

@bGetDeleted: The parameter determines whether to include Web Parts from Web Part Pages that are in the Recycle Bin. When set to 1, the Result Set MUST include Web Parts that are in Web Part Pages that are in the Recycle Bin. When set to 0, the Result Set MUST only return Web Parts that are in Web Part Pages that are not in the Recycle Bin.

@bGetAllUsers: The parameter determines whether to include Web Parts for All Users, or just the current user. When set to 1, the returned Result Set MUST return Web Parts for All Users, including Web Parts in other user's personal views. When set to 0, the returned Result Set MUST only return Web Parts in Shared Views or personal Views of current user.

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

@RequestGuid: The optional request identifier for the current request.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST return the following result set:

3.1.4.46.1 List Web Parts Result Set

This Result Set returns Web Part information associated with the list in Web Part pages, one row per Web Part, ordered by the time the Web Part was added to the Web Part Page. The T-SQL syntax for the result set is as follows:

```sql
SELECT
    tp_ListId uniqueidentifier,
    tp_Type tinyint,
    tp_ID uniqueidentifier,
    tp_Flags int,
    tp_DisplayName nvarchar(255),
    tp_PageUrl nvarchar(260),
    tp_BaseViewId tinyint,
    tp_View varbinary(max),
    tp_Level tinyint,
    tp_ContentTypeId varbinary(512),
    tp_PageUrlId uniqueidentifier,
    tp_AllUserProperties varbinary(max),
    tp_PerUserProperties varbinary(max),
    tp_WebPartIdProperty nvarchar(255),
    tp_Cache varbinary(max)
FROM
    tp_ListWebParts
WHERE
    tp_ListId = @SiteId
    AND tp_Type = @RequestGuid
    AND tp_ID = @bGetAllLevel
    AND tp_Flags = @DocVersion
    AND tp_ContentTypeId = @bGetDeleted
    AND tp_AllUserProperties = @bGetAllUsers
    AND tp_Cache = @SiteId
```

**tp_ListId:** The list identifier (as specified in [MS-WSSFO2], section 2.2.1.5) of the list that contains the Web Part. This MUST be the same as the @ListId parameter.

**Tp_Type:** The Page type (as specified in [MS-WSSFO2], section 2.2.3.14) of the Web Part.

**Tp_ID:** The GUID that identifies the Web Part. This value MUST NOT be NULL.

**Tp_Flags:** The View Flags (as specified in [MS-WSSFO2], section 2.2.2.12) of the Web Part.
**Tp_DisplayName**: The Display Name of the Web Part.


**Tp_BaseViewId**: The base view identifier for the Web Part.

**Tp_View**: The CAML of the Web Part.

**Tp_Level**: The Publishing Level of the Web Part Page.

**Tp_ContentTypeId**: The Content type identifier (as specified in [MS-WSSFO2], section 2.2.1.1). If the Web Part is a list View Web Part, returns the Content type identifier (as specified in [MS-WSSFO2], section 2.2.1.1) of the Content type associated with this view. If the Web Part is not associated with any Content type, then it MUST return 0x. It MUST NOT be NULL.

**Tp_PageUrlId**: The Document identifier (as specified in [MS-WSSFO2], section 2.2.1.2) of the Web Part Page.

**Tp_AllUsersProperties**: A binary payload containing 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.

**Tp_PerUserProperties**: A binary payload containing zero or more personalizable properties on the Web Part. If this value is NULL, then default values will be used for all of the personalizable properties on the Web Part.

**Tp_WebPartIdProperty**: The HTML (HyperText Markup Language) ID attribute of the Web Part. May be NULL. If not NULL, it MUST be unique per Web Part Page.

**Tp_Cache**: Private data cache of the Web Part.

### 3.1.4.47 proc_GetNextWebPartOrder

The proc_GetNextWebPartOrder stored procedure is called to request a Web Part Zone Index that is one larger than the maximum Web Part Zone Index being used by all of the Web Parts in a Web Part Zone. The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_GetNextWebPartOrder(
    @SiteID                   uniqueidentifier,
    @DocID                    uniqueidentifier,
    @ZoneId                   nvarchar(64),
    @NextOrder                int OUTPUT,
    @RequestGuid              uniqueidentifier = NULL OUTPUT
);
```

**@SiteID**: The Site Collection identifier of the site collection which contains the Web Part Page.

**@DocID**: The Document identifier (as specified in [MS-WSSFO2], section 2.2.1.2) of the Web Part Page which contains the Web Part Zone.

**@ZoneId**: The Web Part Zone identifier of the Web Part Zone to calculate the next Web Part Zone Index for.

**@NextOrder**: A Web Part Zone Index that is one larger than the maximum Web Part Zone Index present in the Web Part Zone, returned as an output parameter. This value MUST be 1 if no Web Part zone indexes are present in the Web Part Zone or if @SiteID and @DocID do not reference a
existing Web Part Page or @ZoneId does not reference an existing Web Part Page.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.1.4.48 proc_GetRecycleBinItemEventReceivers

The proc_GetRecycleBinItemEventReceivers stored procedure is called to read the information and event receivers of a specified Recycle Bin item. The T-SQL syntax for the stored procedure is as follows:

```t-sql
PROCEDURE proc_GetRecycleBinItemEventReceivers(
    @SiteId                   uniqueidentifier,
    @WebId                    uniqueidentifier,
    @UserId                   int,
    @DeleteTransactionId      varbinary(16),
    @RequestGuid              uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The site collection identifier of the site collection which contains the specified recycle bin item.

@WebId: The Site identifier (as specified in [MS-WSSFO2], section 2.2.1.11) of the site which contains the recycle bin item.

@UserId: The User identifier (as specified in [MS-WSSFO2], section 2.2.1.13) of the current user.

@DeleteTransactionId: The delete transaction identifier of the recycle bin item.

@RequestGuid: The optional request identifier for the current request.

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 completion.</td>
</tr>
<tr>
<td>1168</td>
<td>No recycle bin item is found for @SiteId and @DeleteTransactionId when @UserId is 0; or no recycle bin item is found for @SiteId, @WebId, @DeleteTransactionId and @UserId when @UserId is not 0; or more than one recycle bin item is found for the given parameters.</td>
</tr>
</tbody>
</table>

Result Sets: MUST return three result sets in the following order when the return code is 0 and MUST NOT return any result sets when the return code is not 0.

3.1.4.48.1 Recycle Bin Item Result Set

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

```t-sql
ItemType               tinyint,
WebUrl                 nvarchar(256),
ListId                 uniqueidentifier,
ListTitle              nvarchar(255),
```
ListItemId               int,
DocId                    uniqueidentifier;

**ItemType:** The type of the recycle bin item. The value MUST be one of the following.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Recycle bin item is a document.</td>
</tr>
<tr>
<td>2</td>
<td>Recycle bin item is a document version.</td>
</tr>
<tr>
<td>3</td>
<td>Recycle bin item is a list item.</td>
</tr>
<tr>
<td>4</td>
<td>Recycle bin item is a list.</td>
</tr>
<tr>
<td>5</td>
<td>Recycle bin item is a folder.</td>
</tr>
<tr>
<td>6</td>
<td>Recycle bin item is a folder with lists.</td>
</tr>
<tr>
<td>7</td>
<td>Recycle bin item is an attachment.</td>
</tr>
<tr>
<td>8</td>
<td>Recycle bin item is a version of a list item.</td>
</tr>
</tbody>
</table>

**WebUrl:** The URL in store-relative form of the site that contained the recycle bin item.

**ListId:** The list identifier (as specified in [MS-WSSFO2], section 2.2.1.5) of the Recycle Bin item list.

**ListTitle:** The title of the recycle bin item list.

**ListItemId:** The list Item identifier (as specified in [MS-WSSFO2], section 2.2.1.6) corresponding to the recycle bin item when the recycle bin item type is 1, 3, 5, 7, or 8. Otherwise ListItemId MUST be NULL when the recycle bin item type is 2, 4, or 6.

**DocId:** The Document identifier (as specified in [MS-WSSFO2], section 2.2.1.2) when the recycle bin item has a corresponding document. Otherwise, DocId is NULL.

### 3.1.4.48.2 List Event Receivers Result Set

This result set contains all the event receivers of the recycle bin item list. The **result set** is defined in the Event Receivers Result Set (as specified in [MS-WSSFO2], section 2.2.5.9).

### 3.1.4.48.3 Site Event Receivers Result Set

This result set contains all the event receivers of the Site that contained the recycle bin item. The **result set** is defined in the Event Receivers Result Set (as specified in [MS-WSSFO2], section 2.2.5.9).

### 3.1.4.49 proc_GetRunnableWorkItems

The proc_GetRunnableWorkItems stored procedure is called to retrieve a restricted set of Work Items for which the Work Item Delivery Date has passed and mark them as In Progress Work Item. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_GetRunnableWorkItems(
    @ProcessingId uniqueidentifier,
```
@ProcessingId: The Work Item Processing identifier of the Work Item Process. The server MUST set to this value the Work Item Processing identifier of any Work Items that it modifies. MUST NOT be NULL.

@SiteId: The Site Collection identifier of the Site Collection. If the parameter is not NULL, then the server MUST only modify and return Work Items associated with this Site Collection. If the parameter is NULL, then the server MUST modify and return Work Items that meet the criteria specified by the other parameters regardless of associated Site Collection.

@WorkItemType: The Work Item type identifier of the Work Item type. The server MUST only modify and return Work Items associated with this Work Item type. MUST NOT be NULL.

@BatchId: The Work Item Batch identifier of the Work Item Batch. If the parameter is not NULL, then the server MUST only modify and return Work Items associated with this Work Item Batch identifier and MUST also mark those Work Items as Throttled Fetch. If the parameter is NULL, then the server MUST modify and return Work Items that meet the criteria specified by the other parameters regardless of associated Work Item Batch identifier.

@MaxFetchSize: The maximum number of Work Items that will be marked as In Progress Work Items. This parameter MUST be non-negative. If the value of the parameter is not 0, then the server MUST limit to the specified value the number of new Work Items it marks as In Progress Work Item. If the value of the parameter is 0, then the server MUST NOT limit the number of items it modifies based on this parameter. MUST NOT be NULL.

@ThrottleThreshold: A limit on the number of work item batches. This parameter MUST be non-negative. This parameter MUST NOT be NULL. The server MUST NOT mark any new items as In Progress Work Item if both:

- The value of this parameter is not 0, and
- The value of this parameter is less than the number of distinct work item batch identifiers in the set of Work Items matching all of the following criteria:
  - Is marked as In Progress Work Item,
  - Is marked as Throttled Fetch,
  - Has an associated Work Item type is given by @WorkItemType, and
  - Has a Work Item Delivery Date that has passed.

@RequestGuid: The optional request identifier for the current request.

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 completion.</td>
</tr>
<tr>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>5</td>
<td>Error: Access denied.</td>
</tr>
</tbody>
</table>

**Result Sets:** MUST return zero or one result sets:

### 3.1.4.49.1 Work Items Result Set

This Result Set returns the Work Items that are marked as In Progress Work Items and match the criteria specified by the parameters. The Result Set is defined in section 2.2.6.5.

### 3.1.4.50 proc_GetRunningWorkBatchCount

The proc_GetRunningWorkBatchCount stored procedure is called to retrieve the count of in progress work item batches. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_GetRunningWorkBatchCount(
    @SiteId                   uniqueidentifier,
    @WorkItemType             uniqueidentifier,
    @RequestGuid              uniqueidentifier = NULL OUTPUT
);
```

**@SiteId**: The Site Collection identifier of the Site Collection. If the parameter is not NULL, then the server MUST only count work item batches associated with this Site Collection. If the parameter is NULL, then the server MUST count work item batches that meet the criteria specified by the other parameters regardless of associated Site Collection.

**@WorkItemType**: The Work Item type identifier of the Work Item type. If the parameter is not NULL, then the server MUST only count work item batches associated with this Work Item type. If the parameter is NULL, then the server MUST count work item batches that meet the criteria specified by the other parameters regardless of Work Item type.

**@RequestGuid**: The optional request identifier for the current request.

**Return Code values**: An integer value which MUST be the count of the work item batches specified by the @SiteId and @WorkItemType parameters.

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

### 3.1.4.51 proc_GetSiteResourceUsage

The proc_GetSiteResourceUsage stored procedure is called to retrieve resource usage values for a site collection. The T-SQL syntax for the stored procedure is as follows:

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

**@SiteId**: The site collection identifier of the site collection for which the resource usage value is to be retrieved.

**Return Code values**: An integer value which MUST be 0.

**Result Sets**: This procedure MUST return the Site Collection Resource Usage Result Set.
3.1.4.51.1 Site Collection Resource Usage Result Set

The Site Collection Resource Usage Result Set contains resource usage values for the site collection specified by the @SiteId parameter. The T-SQL syntax for the result set is as follows:

```
CurrentResourceUsage          float NOT NULL,
AverageResourceUsage          float NOT NULL,
ResourceUsageMaximum          float NOT NULL;
```

**CurrentResourceUsage:** The resource usage value for the specified site collection for the current monitoring interval.

**AverageResourceUsage:** The mean resource usage value for the specified site collection over the available number of monitoring intervals.

**ResourceUsageMaximum:** The maximum level for resource usage for the specified site collection.

3.1.4.52 proc_GetSiteSolutionResourceUsage

The proc_GetSiteSolutionResourceUsage stored procedure is called to retrieve resource usage values for all of the sandboxed solutions in a site collection. The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_GetSiteSolutionResourceUsage (   
    @SiteId            uniqueidentifier,
    @DaysAgo           int
);  
```

**@SiteId:** The site collection identifier of the site collection for which resource usage values are to be retrieved.

**@DaysAgo:** The number of monitoring intervals in the past for which resource usage values are to be retrieved. If NULL, the server MUST retrieve resource usage values for all available periods.

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

**Result Sets:** This procedure MUST return the Site Solution Resource Usage Result Set.

3.1.4.52.1 Site Solution Resource Usage Result Set

This result set contains resource usage values for the sandboxed solutions in the specified site collection for the specified monitoring interval. The result set is specified as the Solution Resource Usage Result Set in section 2.2.6.2.

3.1.4.53 proc_GetSolutionInfo

The proc_GetSolutionInfo stored procedure is called to get information for the execution of a sandboxed solution. The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_GetSolutionInfo (   
    @SiteId           uniqueidentifier,
    @SolutionId       uniqueidentifier
);  
```
@SiteId: The site collection identifier of the site collection in which the sandboxed solution resides.

@SolutionId: The identifier of the sandboxed solution.

Return Code Values: An integer which MUST be 0.

Result Sets: This procedure MUST return the Solution Hash Information Result Set.

3.1.4.53.1 Solution Hash Information Result Set

The Solution Hash Information Result Set contains information about the validation state of a sandboxed solution. The T-SQL syntax for the result set is as follows:

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hash</td>
<td>nvarchar(50) NOT NULL</td>
<td>The implementation-specific hash of the content of the sandboxed solution.</td>
</tr>
<tr>
<td>ValidatorsHash</td>
<td>nvarchar(64) NOT NULL</td>
<td>The implementation-specific hash of the validators that validated the sandboxed solution.</td>
</tr>
<tr>
<td>ValidationErrorMessage</td>
<td>nvarchar(4000)</td>
<td>If the sandboxed solution failed validation, MUST contain the specific error message of the validation failure.</td>
</tr>
<tr>
<td>ResourceQuota</td>
<td>float NOT NULL</td>
<td>The resource usage value for the specified sandboxed solution.</td>
</tr>
<tr>
<td>RecentInvocations</td>
<td>int NOT NULL</td>
<td>The number of invocations of code within this sandboxed solution over the current monitoring interval.</td>
</tr>
<tr>
<td>ResourceQuotaExceeded</td>
<td>int NOT NULL</td>
<td>MUST be 1 if the site collection containing the sandboxed solution has exceeded its maximum level for a resource quota, 0 otherwise.</td>
</tr>
</tbody>
</table>

Must be 1 if the site collection containing the sandboxed solution has exceeded its maximum level for a resource quota, 0 otherwise.

3.1.4.54 proc_GetSolutionResourceQuota

The proc_GetSolutionResourceQuota stored procedure is called to get the resource usage value for a sandboxed solution. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_GetSolutionResourceQuota (
    @SiteId           uniqueidentifier,
    @SolutionId       uniqueidentifier
);
```

@SiteId: The site collection identifier of the site collection in which the sandboxed solution resides.

@SolutionId: The identifier of the sandboxed solution.

Return Code Values: An integer which MUST be 0.
Result Sets: This procedure MUST return the Solution Resource Quota Result Set.

3.1.4.54.1 Solution Resource Quota Result Set

The Solution Resource Quota Result Set contains the resource usage value for a sandboxed solution. The T-SQL syntax for the result set is as follows:

- **ResourceQuota**: The resource usage value for the specified sandboxed solution.
- **RecentInvocations**: The number of invocations of code within this sandboxed solution over the current monitoring interval.
- **ResourceQuotaExceeded**: MUST be 1 if the site collection containing the sandboxed solution has exceeded its maximum level for a resource quota, 0 otherwise.

3.1.4.55 proc_GetSolutionResourceUsage

The proc_GetSolutionResourceUsage stored procedure is called to get the resource usage values for a sandboxed solution in a specified monitoring interval. The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_GetSolutionResourceUsage (
    @SiteId            uniqueidentifier,
    @SolutionId        uniqueidentifier,
    @DaysAgo           int
);
```

- **@SiteId**: The site collection identifier of the site collection in which the sandboxed solution resides.
- **@SolutionId**: The identifier of the sandboxed solution.
- **@DaysAgo**: The number of monitoring intervals in the past for which resource usage values are to be retrieved. If NULL, the server MUST retrieve resource usage values for all available monitoring intervals.

Return Code Values: An integer which MUST be 0.

Result Sets: This procedure MUST return the Solution Resource Usage Result Set.

3.1.4.55.1 Solution Resource Usage Result Set

This result set contains resource usage values for the specified sandboxed solution in the specified site collection for the specified monitoring interval. The result set is specified in section 2.2.6.2.

3.1.4.56 proc_GetSolutionResourceUsageDailyOrdinal

The proc_GetSolutionResourceUsageDailyOrdinal stored procedure is called to get the ordinal for the current monitoring interval for resource usage. The T-SQL syntax for the stored procedure is as follows:
PROCEDURE proc_GetSolutionResourceUsageDailyOrdinal();

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

**Result Sets:** This procedure MUST return the Solution Resource Usage Daily Ordinal Result Set.

### 3.1.4.56.1 Solution Resource Usage Daily Ordinal Result Set

The Solution Resource Usage Daily Ordinal Result set MUST contain one row containing the ordinal for the current monitoring interval for resource usage. The T-SQL syntax for the result set is as follows:

```
DaysAgo int NOT NULL;
```

**DaysAgo:** The ordinal for the current monitoring interval for resource usage.

### 3.1.4.57 proc_GetSolutionsData

The proc_GetSolutionsData stored procedure is called to get information for all sandboxed solutions in a site collection. The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_GetSolutionsData (  
    @SiteId uniqueidentifier
);
```

**@SiteId:** The site collection identifier of the site collection in which the sandboxed solutions reside.

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

**Result Sets:** This procedure MUST return the Solution Data Result Set.

### 3.1.4.57.1 Solution Data Result Set

The Solution Data Result Set contains information about the sandboxed solutions in a site collection. The T-SQL syntax for the result set is as follows:

```
Name nvarchar(128) NOT NULL,
SolutionId uniqueidentifier NOT NULL,
Hash nvarchar(50) NOT NULL,
Status smallint NOT NULL,
HasAssemblies tinyint NOT NULL,
Definitions varbinary(max) NOT NULL;
```

**Name:** The name of the sandboxed solution.

**SolutionId:** The identifier of the sandboxed solution.

**Hash:** The implementation-specific hash of the content of the sandboxed solution.

**Status:** A [Sandboxed Solution Status value](#) corresponding to the status of the sandboxed solution.

**HasAssemblies:** MUST be 1 if the sandboxed solution contains assemblies. 0 otherwise.
Definitions: The implementation-specific serialization of the feature definitions for the sandboxed solution.

3.1.4.58 proc_GetWFTemplatesLastModifiedForWeb

The proc_GetWFTemplatesLastModifiedForWeb stored procedure is called to retrieve modification information about document libraries contained in a site collection and site. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_GetWFTemplatesLastModifiedForWeb (
    @SiteId uniqueidentifier,
    @WebId uniqueidentifier,
    @SiteRootLastModified datetime OUTPUT,
    @CurWebLastModified datetime OUTPUT,
    @SiteRootItemCount int OUTPUT,
    @CurWebItemCount int OUTPUT,
    @RequestGuid uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The Site Collection identifier of the Site Collection which contains the document libraries. This value MUST NOT be NULL.

@WebId: The Site identifier (as specified in [MS-WSSFO2], section 2.2.1.11) of the Site which contains the document libraries. This value MUST NOT be NULL.

@SiteRootLastModified: The protocol server MUST ignore the input value of this parameter. On output, the protocol server MUST set this value to the latest modification time of any document library based on template 122 (No Code Public) contained in the Site Collection specified by @SiteId, or NULL if there are no document libraries matching the criteria.

@CurWebLastModified: The protocol server MUST ignore the input value of this parameter. On output, the protocol server MUST set this value to the latest modification time of any document library based on template 117 (No Code Workflows) contained in the Site specified by @WebId, or NULL if there are no document libraries matching the criteria.

@SiteRootItemCount: The protocol server MUST ignore the input value of this parameter. On output, the protocol server MUST set this value to the total count of items contained in all document libraries based on template 122 (No Code Public) contained in the Site Collection specified by @SiteId, or NULL if there are no document libraries matching the criteria.

@CurWebItemCount: The protocol server MUST ignore the input value of this parameter. On output, the protocol server MUST set this value to the total count of items contained in all document libraries based on template 117 (No Code Workflows) contained in the Site specified by @WebId, or NULL if there are no document libraries matching the criteria.

@RequestGuid: The optional request identifier for the current request.

3.1.4.59 proc_GetWorkflowAssociations

The proc_GetWorkflowAssociations stored procedure is called to get a set of Workflow associations. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_GetWorkflowAssociations (
    @SiteId uniqueidentifier,
    @WebId uniqueidentifier,
);
```
@Id        uniqueidentifier,
@ListId    uniqueidentifier,
@ContentTypeId  varbinary(512),
@RequestGuid uniqueidentifier = NULL OUTPUT
);

@SiteId: The Site Collection identifier of the Site Collection which contains the Workflow associations.

@WebId: The Site identifier (as specified in [MS-WSSFO2], section 2.2.1.11) of the Site which contains the Workflow associations. If this value is NULL, the server MUST include all Sites.

@Id: The Workflow association identifier of the Workflow association. If this value is not null, the server MUST ignore @WebId, @ListId and @ContentTypeId, and the Result Set MUST contain exactly one row containing the Workflow association specified by @Id.

@ListId: The list identifier of the list the Workflow associations are associated with. If this value is NULL, the server MUST include all lists.

@ContentTypeId: The Content type identifier (as specified in [MS-WSSFO2], section 2.2.1.1) of the Content type the Workflow associations are associated with. If this value is NULL, the server MUST include all Content types.

@RequestGuid: The optional request identifier for the current request.

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

Result Sets: MUST return the Workflow Associations Result Set.

3.1.4.59.1 Workflow Associations Result Set

The Result Set is defined in section 2.2.6.5.

3.1.4.60 proc_GetWorkflowDataForItem

The proc_GetWorkflowDataForItem stored procedure is called to obtain data about Workflows and Workflow associations. The T-SQL syntax for the stored procedure is as follows:

PROCEDURE proc_GetWorkflowDataForItem(
    @SiteId                  uniqueidentifier,
    @WebId                   uniqueidentifier,
    @ListId                  uniqueidentifier,
    @ItemID                  int,
    @ContentTypeId           varbinary(512),
    @gwfdi                   int = 0xF,
    @InclusiveFilterState    int = 0xFFFFFFFF,
    @ExclusiveFilterState    int = 0,
    @RequestGuid             uniqueidentifier = NULL OUTPUT
);

@SiteId: The Site Collection identifier of the Site Collection which contains the Workflows and Workflow associations.

@WebId: The Site identifier (as specified in [MS-WSSFO2], section 2.2.1.11) of the Site which contains the Workflows and Workflow associations.
@ListId: The list identifier of the list which contains the list Items the Workflows were created for. If this value is NULL, the server MUST include all lists.

@ItemId: The list Item identifier (as specified in [MS-WSSFO2], section 2.2.1.6) of the list Item the Workflows were created for. If @ListId is NULL, @Itemid MUST be NULL. If this value is NULL, the server MUST include all list items.

@ContentTypeId: The Content type identifier (as specified in [MS-WSSFO2], section 2.2.1.1) with which the Workflows are associated.

@gwfdi: A bitmask which determines which Result Sets are returned. MUST contain zero or more of the flags listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The server MUST return the Workflow Associations Result Set for the list specified by @ListId. @ListId MUST NOT be NULL.</td>
</tr>
<tr>
<td>2</td>
<td>The server MUST return the Workflow Associations Result Set for the Content type specified by @ContentTypeId. @ContentTypeId MUST NOT be NULL.</td>
</tr>
<tr>
<td>4</td>
<td>The server MUST return the List Item Workflows Result Set for the list Item specified by @ItemId. @ListId and @ItemId MUST NOT be NULL.</td>
</tr>
</tbody>
</table>

@InclusiveFilterState: A workflow internal state bitmask. The server MUST include only Workflows that have at least one internal state bit flag in common with @InclusiveFilterState (that is, Workflow.InternalState & @InclusiveFilterState <> 0) in the List Item Workflows Result Set.

@ExclusiveFilterState: A workflow internal state bitmask. The server MUST exclude any Workflows that have any internal state bit flags in common with @ExclusiveFilterState (that is, Workflow.InternalState & @ExclusiveFilterState <> 0) from the List Item Workflows Result Set.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: An integer which MUST be 0.

Result Set: MUST return 0, 1, or 2 Workflow Associations Result Sets and 0 or 1 List Item Workflows Result Set based on the @gwfdi parameter, ordered from the lowest flag (1) to the highest (4).

3.1.4.60.1 Workflow Associations Result Set

The Result Set is defined in 2.2.6.5.

3.1.4.60.2 List Item Workflows Result Set

If the List Item Workflows Result Set is returned, the InstanceData and ProcessingId columns MUST be NULL and the InstanceDataSize column MUST contain the value 0. The Result Set is defined in section 2.2.6.1.

3.1.4.61 proc_GetWorkItems

The proc_GetWorkItems stored procedure is called to retrieve a set of Work Items that meet the specified criteria. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_GetWorkItems(
```
@SiteId: The Site Collection identifier of the Site Collection. The server MUST only return Work Items associated with this Site Collection identifier. MUST NOT be NULL.

@ParentId: The Work Item Parent identifier of the Work Item. If @WorkItemId is NULL and this parameter is not NULL, then the server MUST only return Work Items which have this Work Item Parent identifier. If this parameter is NULL, then the server MUST return Work Items that meet the criteria specified by the other parameters, regardless of the value of their Work Item Parent identifier.

@WorkItemType: The Work Item type identifier of the Work Item type. If @WorkItemId is NULL, then this parameter MUST NOT be NULL, and the server MUST only return Work Items associated with this Work Item type. If @WorkItemId is not NULL, then the server MUST return Work Items that meet the criteria specified by the other parameters, regardless of associated Work Item type.

@BatchId: The Work Item Batch identifier of the work item batch. If @WorkItemId is NULL and the parameter is not NULL, then the server MUST only return Work Items associated with this Work Item Batch. If this parameter is NULL, then the server MUST return Work Items that meet the criteria specified by the other parameters, regardless of the associated work item batch.

@WorkItemId: The work item identifier. If the parameter is not NULL, then the server MUST restrict the returned Work Item to have a Work Item identifier matching the parameter and associated with the Site Collection indicated by @SiteId. If this parameter is NULL, then the server MUST return Work Items that meet the criteria specified by the other parameters, regardless of associated work item identifier.

@RequestGuid: The optional request identifier for the current request.

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

Result Sets: MUST return the following result set:

3.1.4.61.1 Single Work Item Result Set

This Result Set returns a single Work Item that meets the criteria specified by the parameters when @WorkItemId is not NULL. The Result Set is defined in section 2.2.6.5.

3.1.4.61.2 Multiple Work Items Result Set

This Result Set returns the Work Items that meet the criteria specified by the parameters when @WorkItemId is NULL. The Result Set is defined in section 2.2.6.5.

3.1.4.62 proc_InsertContextEventReceiver

The proc_InsertContextEventReceiver stored procedure is called to create a new Event Receiver and, optionally, create an additional Event Receiver that the new Event Receiver will be registered against. The T-SQL syntax for the stored procedure is as follows:
PROCEDURE proc_InsertContextEventReceiver(
    @Id                      uniqueidentifier,
    @Name                    nvarchar(256),
    @SiteId                  uniqueidentifier,
    @WebId                   uniqueidentifier,
    @ParentHostId            uniqueidentifier,
    @ParentHostType          int,
    @Synchronization         int,
    @Type                    int,
    @SequenceNumber          int,
    @Assembly                nvarchar(256),
    @Class                   nvarchar(256),
    @Data                    nvarchar(256),
    @Filter                  nvarchar(256),
    @Credential              int,
    @ContextHostType         int,
    @ContextObjectItemId     int,
    @ContextObjectUrl        nvarchar(260),
    @ContextType             uniqueidentifier,
    @ContextEventType       uniqueidentifier,
    @ContextId               uniqueidentifier,
    @ContextObjectId         uniqueidentifier,
    @ContextCollectionId     uniqueidentifier,
    @RequestGuid             uniqueidentifier = NULL OUTPUT
);
<table>
<thead>
<tr>
<th>Value</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>required to run the task using the same thread that is processing the request whose action triggered the event.</td>
</tr>
</tbody>
</table>

@Type: The type of the Event Receiver. @Type MUST be a value of the Event Receiver type (as specified in [MS-WSSFO2], section 2.2.3.6).

@SequenceNumber: The sequence number (1) of the event receiver. @SequenceNumber MUST be greater than or equal to zero and less than or equal to 65535.

@Assembly: The Assembly Name strong name of the assembly that contains the Event Receiver. This value MUST NOT be NULL.

@Class: The fully qualified class name of the Event Receiver in the assembly. This value MUST NOT be NULL.

@Data: Additional data to be passed to the Event Receiver.

@Filter: Reserved. @Filter MUST be NULL.

@Credential: Reserved. @Credential MUST be zero.

@ContextHostType: The type of the event host of the event receiver. The value MUST be one of Event Host type (as specified in [MS-WSSFO2], section 2.2.3.5).

@ContextObjectId: The context object identifier of the Event Host for which an Event Receiver is registered.

@ContextObjectUrl: Reserved. @ContextObjectUrl MUST be NULL.

@ContextType: The context type identifier of the event receiver.

@ContextEventType: Reserved. @ContextEventType MUST be NULL.

@ContextId: The context identifier of the event receiver.

@RequestGuid: The optional request identifier for the current request.

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 completion.</td>
</tr>
<tr>
<td>30</td>
<td>An error occurred.</td>
</tr>
<tr>
<td>87</td>
<td>@ContextCollectionId is NULL and no Event Receivers were inserted or the insertion of Event Receivers failed.</td>
</tr>
</tbody>
</table>

Result Sets: MUST NOT return any result sets.
3.1.4.63 proc_InsertDocEventReceiver

The proc_InsertDocEventReceiver stored procedure is called to register an event receiver for a specified document. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_InsertDocEventReceiver(
    @DocUrl                   nvarchar(260),
    @Id                       uniqueidentifier,
    @Name                     nvarchar(256),
    @SiteId                   uniqueidentifier,
    @WebId                    uniqueidentifier,
    @ItemId       int,
    @Synchronization          int,
    @Type                     int,
    @SequenceNumber           int,
    @Assembly                 nvarchar(256),
    @Class                    nvarchar(256),
    @SolutionId               uniqueidentifier,
    @Data                     nvarchar(256),
    @Filter                   nvarchar(256),
    @Credential               int,
    @RequestGuid              uniqueidentifier = NULL OUTPUT
);
```

@DocUrl: The URL in store-relative form of the document.

@Id: The Event Receiver identifier (as specified in [MS-WSSFO2], section 2.2.1.3) of the event receiver. This value MUST NOT be NULL.

@Name: The name of the event receiver. This value MUST NOT be NULL.

@SiteId: The site collection identifier of the site collection which contains the document.

@WebId: The Site identifier (as specified in [MS-WSSFO2], section 2.2.1.11) of the site which contains the document.

@ItemId: Reserved. @ItemId MUST be 0.

@Synchronization: Specifies the synchronicity of the event receiver and the action triggering the event. The value MUST be an integer which is listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Default</td>
<td>For before event receivers, the server MUST run the event receiver synchronously. For after event receivers, the server is not required to run the event synchronously.</td>
</tr>
<tr>
<td>1</td>
<td>Synchronous</td>
<td>The server MUST run the event receiver using the same thread that is processing the request whose action triggered the event.</td>
</tr>
<tr>
<td>2</td>
<td>Asynchronous</td>
<td>The server MUST queue the task of running the event receiver. The server is not required to run the task using the same thread that is processing the request whose action triggered the event.</td>
</tr>
</tbody>
</table>

@Type: The type of the event receiver. @Type MUST be one of Event Receiver type (as specified in [MS-WSSFO2], section 2.2.3.6).
@SequenceNumber: The sequence number (1) of the event receiver. @SequenceNumber MUST be greater than or equal to zero and less than or equal to 65535.

@Assembly: The assembly name of the implementation of the event receiver. This value MUST NOT be NULL.

@Class: The fully qualified class name of the implementation of the event receiver. This value MUST NOT be NULL.

@SolutionId: The identifier of the sandboxed solution that is the source of the event receiver.

@Data: Additional data persisted on behalf of the event receiver implementation to be passed to the event receiver.

@Filter: Reserved. @Filter MUST be NULL.

@Credential: Reserved. @Credential MUST be zero.

@RequestGuid: The optional request identifier for the current request.

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>Insertion succeeded.</td>
</tr>
<tr>
<td>3</td>
<td>The document identified by @DocUrl is not found in the site identified by @WebId in the site collection identified by @SiteId.</td>
</tr>
<tr>
<td>87</td>
<td>The insertion failed.</td>
</tr>
</tbody>
</table>

Result Sets: MUST NOT return any result sets.

3.1.4.64 proc_InsertEventReceiver

The proc_InsertEventReceiver stored procedure is called to register an event receiver for a specified event host. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_InsertEventReceiver(
    @Id uniqueidentifier,
    @Name nvarchar(256),
    @SiteId uniqueidentifier,
    @WebId uniqueidentifier,
    @HostId uniqueidentifier,
    @HostType int,
    @ItemId int,
    @DirName nvarchar(256),
    @LeafName nvarchar(128),
    @Synchronization int,
    @Type int,
    @SequenceNumber int,
    @Assembly nvarchar(256),
    @Class nvarchar(256),
    @SolutionId uniqueidentifier,
    @Data nvarchar(256),
    @Filter nvarchar(256),
    @SourceId varbinary(512),
    @SourceType int,
)```
@Id: The Event Receiver identifier (as specified in [MS-WSSFO2], section 2.2.1.3) of the event receiver. This value MUST NOT be NULL.

@Name: The name of the event receiver. This value MUST NOT be NULL.

@SiteId: The Site Collection identifier of the site collection which contains the event host. This value MUST NOT be NULL.

@WebId: The Site identifier (as specified in [MS-WSSFO2], section 2.2.1.11) of the site which contains the event host. If there is no associated Site, then this value MUST be the empty GUID. This value MUST NOT be NULL.

@HostId: The event host identifier of the event host of the event receiver. This value MUST NOT be NULL.

@HostType: The type of the event host of the event receiver. @HostType MUST be one of Event Host type (as specified in [MS-WSSFO2], section 2.2.3.5).

@ItemId: Reserved. @ItemId MUST be 0.

@DirName: Reserved. @DirName MUST be NULL.

@LeafName: Reserved. @LeafName MUST be NULL.

@Synchronization: Specifies the synchronicity of the event receiver and the action triggering the event. The value MUST be an integer which is listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Default</td>
<td>For before event receivers, the server MUST run the event receiver synchronously. For after event receivers, the server is not required to run the event synchronously.</td>
</tr>
<tr>
<td>1</td>
<td>Synchronous</td>
<td>The server MUST run the event receiver synchronously.</td>
</tr>
<tr>
<td>2</td>
<td>Asynchronous</td>
<td>The server MUST queue the task of running the event receiver. The server is not required to run the task synchronously.</td>
</tr>
</tbody>
</table>

@Type: The type of the event receiver. @Type MUST be one of Event Receiver type (as specified in [MS-WSSFO2], section 2.2.3.6).

@SequenceNumber: The sequence number (1) of the event receiver. @SequenceNumber MUST greater than or equal to zero and less than or equal to 65535.

@Assembly: The assembly name of the implementation of the event receiver. This value MUST NOT be NULL.
@Class: The fully qualified class name of the implementation of the event receiver. This value MUST NOT be NULL.

@SolutionId: The identifier of the sandboxed solution.

@Data: Additional data persisted on behalf of the event receiver implementation to be passed to the event receiver.

@Filter: Reserved. @Filter MUST be NULL.

@SourceId: The event receiver source identifier of the event receiver. This is the Feature identifier (as specified in [MS-WSSFO2], section 2.2.1.4) of the feature if the event receiver is added via a feature. This is the Content type identifier (as specified in [MS-WSSFO2], section 2.2.1.1) of the content type if the event receiver is added via a content type. Otherwise the event receiver source identifier MUST be NULL.

@SourceType: The Event Receiver Source type of the event receiver. @SourceType MUST be one of the Event Receiver Source Type values.

@Credential: Reserved. @Credential MUST be zero.

@ContextType: The context type identifier of the event receiver.

@ContextEventType: Reserved. @ContextEventType MUST be NULL.

@ContextId: The context identifier of the event receiver.

@ContextObjectId: The context object identifier for the Event Host of the event receiver.

@ContextCollectionId: The context collection identifier of the event receiver.

@RequestGuid: The optional request identifier for the current request.

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>Insertion succeeded.</td>
</tr>
<tr>
<td>30</td>
<td>An error occurred.</td>
</tr>
<tr>
<td>87</td>
<td>Insertion failed.</td>
</tr>
</tbody>
</table>

Result Sets: MUST NOT return any result sets.

3.1.4.65  proc_LogSolutionResourceUsage20

The proc_LogSolutionResourceUsage20 stored procedure is called to log to the immediate solution resource usage log up to 20 resource usage measurements for sandboxed solutions within a given site collection. The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_LogSolutionResourceUsage20 (  
    @SiteId uniqueidentifier,  
    @SolutionId01 uniqueidentifier = null,  
    @ResourceId01 uniqueidentifier = null,  
    @StartTime01 datetime = null,  
    @EndTime01 datetime = null,
```

---

[MS-WSSPROG2] — v20120630
Windows SharePoint Services: Content Database Programmability Extensions Communications Version 2 Protocol Specification

Copyright © 2012 Microsoft Corporation.

Release: July 16, 2012
| @SampleCount01 | int = null, |
| @ResourceUsage01 | float = null, |
| @SolutionId02 | uniqueidentifier = null, |
| @ResourceId02 | uniqueidentifier = null, |
| @StartTime02 | datetime = null, |
| @EndTime02 | datetime = null, |
| @SampleCount02 | int = null, |
| @ResourceUsage02 | float = null, |
| @SolutionId03 | uniqueidentifier = null, |
| @ResourceId03 | uniqueidentifier = null, |
| @StartTime03 | datetime = null, |
| @EndTime03 | datetime = null, |
| @SampleCount03 | int = null, |
| @ResourceUsage03 | float = null, |
| @SolutionId04 | uniqueidentifier = null, |
| @ResourceId04 | uniqueidentifier = null, |
| @StartTime04 | datetime = null, |
| @EndTime04 | datetime = null, |
| @SampleCount04 | int = null, |
| @ResourceUsage04 | float = null, |
| @SolutionId05 | uniqueidentifier = null, |
| @ResourceId05 | uniqueidentifier = null, |
| @StartTime05 | datetime = null, |
| @EndTime05 | datetime = null, |
| @SampleCount05 | int = null, |
| @ResourceUsage05 | float = null, |
| @SolutionId06 | uniqueidentifier = null, |
| @ResourceId06 | uniqueidentifier = null, |
| @StartTime06 | datetime = null, |
| @EndTime06 | datetime = null, |
| @SampleCount06 | int = null, |
| @ResourceUsage06 | float = null, |
| @SolutionId07 | uniqueidentifier = null, |
| @ResourceId07 | uniqueidentifier = null, |
| @StartTime07 | datetime = null, |
| @EndTime07 | datetime = null, |
| @SampleCount07 | int = null, |
| @ResourceUsage07 | float = null, |
| @SolutionId08 | uniqueidentifier = null, |
| @ResourceId08 | uniqueidentifier = null, |
| @StartTime08 | datetime = null, |
| @EndTime08 | datetime = null, |
| @SampleCount08 | int = null, |
| @ResourceUsage08 | float = null, |
| @SolutionId09 | uniqueidentifier = null, |
| @ResourceId09 | uniqueidentifier = null, |
| @StartTime09 | datetime = null, |
| @EndTime09 | datetime = null, |
| @SampleCount09 | int = null, |
| @ResourceUsage09 | float = null, |
| @SolutionId10 | uniqueidentifier = null, |
| @ResourceId10 | uniqueidentifier = null, |
| @StartTime10 | datetime = null, |
| @EndTime10 | datetime = null, |
| @SampleCount10 | int = null, |
| @ResourceUsage10 | float = null, |
| @SolutionId11 | uniqueidentifier = null, |
| @ResourceId11 | uniqueidentifier = null, |
| @StartTime11 | datetime = null, |
@EndTime11          datetime = null,
@SampleCount11       int = null,
@ResourceUsage11     float = null,
@SolutionId12       uniqueidentifier = null,
@ResourceId12       uniqueidentifier = null,
@StartTime12        datetime = null,
@EndTime12          datetime = null,
@SampleCount12       int = null,
@ResourceUsage12     float = null,
@SolutionId13       uniqueidentifier = null,
@ResourceId13       uniqueidentifier = null,
@StartTime13        datetime = null,
@EndTime13          datetime = null,
@SampleCount13       int = null,
@ResourceUsage13     float = null,
@SolutionId14       uniqueidentifier = null,
@ResourceId14       uniqueidentifier = null,
@StartTime14        datetime = null,
@EndTime14          datetime = null,
@SampleCount14       int = null,
@ResourceUsage14     float = null,
@SolutionId15       uniqueidentifier = null,
@ResourceId15       uniqueidentifier = null,
@StartTime15        datetime = null,
@EndTime15          datetime = null,
@SampleCount15       int = null,
@ResourceUsage15     float = null,
@SolutionId16       uniqueidentifier = null,
@ResourceId16       uniqueidentifier = null,
@StartTime16        datetime = null,
@EndTime16          datetime = null,
@SampleCount16       int = null,
@ResourceUsage16     float = null,
@SolutionId17       uniqueidentifier = null,
@ResourceId17       uniqueidentifier = null,
@StartTime17        datetime = null,
@EndTime17          datetime = null,
@SiteId: The site collection identifier of the site collection for this resource usage measurement.

The next six parameters are duplicated 20 times, with each set of parameters referring to a resource usage measurement to be logged. Each instance of these individual parameter names is differentiated by a suffix with a value of 01 through 20 inclusive, which replaces the placeholder "#" symbol shown following. Each group is optional. To signify that a group is to be ignored, the client MUST specify a @ResourceId value of NULL, in which case the server MUST ignore the other parameters with that suffix value.

@SolutionId#: The identifier of the sandboxed solution for this resource usage measurement.

@ResourceId#: The identifier of the monitored resource measure for this resource usage measurement.

@StartTime#: The beginning of the time interval, in the local time zone of the front-end Web server, corresponding to this resource usage measurement.

@EndTime#: The end of the time interval, in the local time zone of the front-end Web server, corresponding to this resource usage measurement.

@SampleCount#: The number of sample points taken for this resource usage measurement.

@ResourceUsage#: The resource usage measurement for the given sandboxed solution, monitored resource measure, and time interval.

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

Result Sets: MUST NOT return any result sets.

3.1.4.66 proc_LogSolutionResourceUsageDaily20

The proc_LogSolutionResourceUsageDaily20 stored procedure is called to log to the daily solution resource usage log up to 20 aggregated daily resource usage measurements for sandboxed solutions within a given site collection. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_LogSolutionResourceUsageDaily20 (
    @SiteId01    uniqueidentifier = null,
    @SolutionId01 uniqueidentifier = null,
    @ResourceId01 uniqueidentifier = null,
    @SampleCount01 int = null,
    @ResourceUsage01 float = null,
    @SiteId02    uniqueidentifier = null,
    @SolutionId02 uniqueidentifier = null,
    @ResourceId02 uniqueidentifier = null,
    @SampleCount02 int = null,
    @ResourceUsage02 float = null,
    @SiteId03    uniqueidentifier = null,
    @SolutionId03 uniqueidentifier = null,
    @ResourceId03 uniqueidentifier = null,
    @SampleCount03 int = null,
    @ResourceUsage03 float = null,
    @SiteId04    uniqueidentifier = null,
    @SolutionId04 uniqueidentifier = null,
    @ResourceId04 uniqueidentifier = null,
    @SampleCount04 int = null,
    @ResourceUsage04 float = null,
    @SiteId05    uniqueidentifier = null,
    @SolutionId05 uniqueidentifier = null,

```
@ResourceId05 uniqueidentifier = null,
@ResourceCount05 int = null,
@ResourceUsage05 float = null,
@SiteId06 uniqueidentifier = null,
@SolutionId06 uniqueidentifier = null,
@ResourceId06 uniqueidentifier = null,
@ResourceCount06 int = null,
@ResourceUsage06 float = null,
@SiteId07 uniqueidentifier = null,
@SolutionId07 uniqueidentifier = null,
@ResourceId07 uniqueidentifier = null,
@ResourceCount07 int = null,
@ResourceUsage07 float = null,
@SiteId08 uniqueidentifier = null,
@SolutionId08 uniqueidentifier = null,
@ResourceId08 uniqueidentifier = null,
@ResourceCount08 int = null,
@ResourceUsage08 float = null,
@SiteId09 uniqueidentifier = null,
@SolutionId09 uniqueidentifier = null,
@ResourceId09 uniqueidentifier = null,
@ResourceCount09 int = null,
@ResourceUsage09 float = null,
@SiteId10 uniqueidentifier = null,
@SolutionId10 uniqueidentifier = null,
@ResourceId10 uniqueidentifier = null,
@ResourceCount10 int = null,
@ResourceUsage10 float = null,
@SiteId11 uniqueidentifier = null,
@SolutionId11 uniqueidentifier = null,
@ResourceId11 uniqueidentifier = null,
@ResourceCount11 int = null,
@ResourceUsage11 float = null,
@SiteId12 uniqueidentifier = null,
@SolutionId12 uniqueidentifier = null,
@ResourceId12 uniqueidentifier = null,
@ResourceCount12 int = null,
@ResourceUsage12 float = null,
@SiteId13 uniqueidentifier = null,
@SolutionId13 uniqueidentifier = null,
@ResourceId13 uniqueidentifier = null,
@ResourceCount13 int = null,
@ResourceUsage13 float = null,
@SiteId14 uniqueidentifier = null,
@SolutionId14 uniqueidentifier = null,
@ResourceId14 uniqueidentifier = null,
@ResourceCount14 int = null,
@ResourceUsage14 float = null,
@SiteId15 uniqueidentifier = null,
@SolutionId15 uniqueidentifier = null,
@ResourceId15 uniqueidentifier = null,
@ResourceCount15 int = null,
@ResourceUsage15 float = null,
The next five parameters are duplicated 20 times, with each set of parameters referring to an aggregated daily resource usage measurement to be logged. Each instance of these individual parameter names is differentiated by a suffix with a value of 01 through 20 inclusive, which replaces the placeholder "#" symbol shown following. Each group is optional. To signify that a group is to be ignored, the client MUST specify a @ResourceId# value of NULL, in which case the server MUST ignore the other parameters with that suffix value.

@SiteId#: The site collection identifier of the site collection for this resource usage measurement.

@SolutionId#: The identifier of the sandboxed solution for this resource usage measurement.

@ResourceId#: The identifier of the monitored resource measure for this resource usage measurement.

@SampleCount#: The number of sample points taken for this resource usage measurement.

@ResourceUsage#: The resource usage measurement for the given site collection, sandboxed solution and monitored resource measure.

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

Result Sets: MUST NOT return any result sets.

3.1.4.67 proc_LogSolutionResourceUsageWindowed20

The proc_LogSolutionResourceUsageWindowed20 stored procedure is called to log to the windowed solution resource usage log up to 20 resource usage measurements for sandboxed solution. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_LogSolutionResourceUsageWindowed20 (  
    @DaysAgo                  int,  
    @SiteId01                 uniqueidentifier = null,
    @SolutionId01             uniqueidentifier = null,
    @ResourceId01             uniqueidentifier = null,
    @StartTime01              datetime = null,
```
@EndTime01          datetime = null,
@SampleCount01       int = null,
@ResourceUsage01     float = null,
@SiteId02           uniqueidentifier = null,
@SolutionId02       uniqueidentifier = null,
@ResourceId02       uniqueidentifier = null,
@StartTime02        datetime = null,
@EndTime02          datetime = null,
@SampleCount02       int = null,
@ResourceUsage02     float = null,
@SiteId03           uniqueidentifier = null,
@SolutionId03       uniqueidentifier = null,
@ResourceId03       uniqueidentifier = null,
@StartTime03        datetime = null,
@EndTime03          datetime = null,
@SampleCount03       int = null,
@ResourceUsage03     float = null,
@SiteId04           uniqueidentifier = null,
@SolutionId04       uniqueidentifier = null,
@ResourceId04       uniqueidentifier = null,
@StartTime04        datetime = null,
@EndTime04          datetime = null,
@SampleCount04       int = null,
@SiteId10 uniqueidentifier = null,
@SolutionId10 uniqueidentifier = null,
@ResourceId10 uniqueidentifier = null,
@StartTime10 datetime = null,
@EndTime10 datetime = null,
@SampleCount10 int = null,
@ResourceUsage10 float = null,
@SiteId11 uniqueidentifier = null,
@SolutionId11 uniqueidentifier = null,
@ResourceId11 uniqueidentifier = null,
@StartTime11 datetime = null,
@EndTime11 datetime = null,
@SampleCount11 int = null,
@ResourceUsage11 float = null,
@SiteId12 uniqueidentifier = null,
@SolutionId12 uniqueidentifier = null,
@ResourceId12 uniqueidentifier = null,
@StartTime12 datetime = null,
@EndTime12 datetime = null,
@SampleCount12 int = null,
@ResourceUsage12 float = null,
@SiteId13 uniqueidentifier = null,
@SolutionId13 uniqueidentifier = null,
@ResourceId13 uniqueidentifier = null,
@StartTime13 datetime = null,
@EndTime13 datetime = null,
@SampleCount13 int = null,
@ResourceUsage13 float = null,
@SiteId14 uniqueidentifier = null,
@SolutionId14 uniqueidentifier = null,
@ResourceId14 uniqueidentifier = null,
@StartTime14 datetime = null,
@EndTime14 datetime = null,
@SampleCount14 int = null,
@ResourceUsage14 float = null,
@SiteId15 uniqueidentifier = null,
@SolutionId15 uniqueidentifier = null,
@ResourceId15 uniqueidentifier = null,
@StartTime15 datetime = null,
@EndTime15 datetime = null,
@SampleCount15 int = null,
@ResourceUsage15 float = null,
@SiteId16 uniqueidentifier = null,
@SolutionId16 uniqueidentifier = null,
@ResourceId16 uniqueidentifier = null,
@StartTime16 datetime = null,
@EndTime16 datetime = null,
@SampleCount16 int = null,
@ResourceUsage16 float = null,
@SiteId17 uniqueidentifier = null,
@SolutionId17 uniqueidentifier = null,
@ResourceId17 uniqueidentifier = null,
@StartTime17 datetime = null,
@EndTime17 datetime = null,
@SampleCount17 int = null,
@ResourceUsage17 float = null,
@DaysAgo: The ordinal for the monitoring interval associated with these resource usage measurements. If NULL, the server MUST use the current ordinal for the monitoring interval for resource usage.

The next seven parameters are duplicated 20 times, with each set of parameters referring to a resource usage measurement to be logged. Each instance of these individual parameter names is differentiated by a suffix with a value of 01 through 20 inclusive, which replaces the placeholder "#" symbol shown following. Each group is optional. To signify that a group is to be ignored, the client MUST specify a @ResourceId# value of NULL, in which case the server MUST ignore the other parameters with that suffix value.

@SiteId#: The site collection identifier of the site collection for this resource usage measurement.

@SolutionId#: The identifier of the sandboxed solution for this resource usage measurement.

@ResourceId#: The identifier of the monitored resource measure for this resource usage measurement.

@StartTime#: The beginning of the time interval, in the local time zone of the front-end Web server, corresponding to this resource usage measurement.

@EndTime#: The end of the time interval, in the local time zone of the front-end Web server, corresponding to this resource usage measurement.

@SampleCount#: The number of sample points taken for this resource usage measurement.

@ResourceUsage#: The resource usage measurement for the given sandboxed solution, monitored resource measure, and time interval.

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

Result Sets: MUST NOT return any result sets.
3.1.4.68 proc_ProcessSolutionResourceUsageLogData

The proc_ProcessSolutionResourceUsageLogData stored procedure is called to read resource usage measurements from the immediate solution resource usage log in batches for aggregation. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_ProcessSolutionResourceUsageLogData (  
    @IdStart bigint
);
```

@IdStart: The identifier of the resource usage measurement for the record with which this batch is to start. If NULL, the server MUST send the resource usage measurements with the largest identifiers, up to 1001 resource usage measurements. Otherwise, the server MUST send the resource usage measurements with the largest identifiers strictly less than @IdStart, up to 1001 resource usage measurements.

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

Result Sets: This procedure MUST return the Solution Resource Usage Log Processing Result Set.

3.1.4.68.1 Solution Resource Usage Log Processing Result Set

This result set MUST contain from 0 up to 1001 rows containing resource usage measurements from the immediate solution resource usage log. The result set is defined in the Solution Resource Usage Processing Result Set (as specified in section 2.2.6.3).

3.1.4.69 proc_ProcessSolutionResourceUsageWindowedData

The proc_ProcessSolutionResourceUsageWindowedData stored procedure is called to read resource usage measurements from the windowed solution resource usage log in batches for aggregation. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_ProcessSolutionResourceUsageWindowedData (  
    @IdStart bigint,
    @DaysAgo int
);
```

@IdStart: The identifier for the resource usage measurement record with which this batch is to start. If NULL, the server MUST send the resource usage measurements with the largest identifiers, up to 1001 resource usage measurements. Otherwise, the server MUST send the resource usage measurements with the largest identifiers strictly less than @IdStart, up to 1001 resource usage measurements.

@DaysAgo: The ordinal for the monitoring interval for resource usage for which records are being retrieved.

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

Result Sets: This procedure MUST return the Windowed Solution Resource Usage Processing Result Set.
3.1.4.69.1 Windowed Solution Resource Usage Processing Result Set

This result set MUST contain 0 to 1001 rows containing resource usage measurements from the windowed solution resource usage log. The **result set** is defined in the Solution Resource Usage Processing Result Set (as specified in section 2.2.6.3).

3.1.4.70 proc_ProvisionWebPart

The proc_ProvisionWebPart stored procedure is called to add a new Web Part to a Web Part Page. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_ProvisionWebPart(
    @SiteId                    uniqueidentifier,
    @DocID                     uniqueidentifier,
    @WebPartID                 uniqueidentifier,
    @Level                     tinyint,
    @IsIncluded                bit,
    @FrameState                tinyint,
    @ZoneID                    nvarchar(64),
    @PartOrder                 int,
    @Source                    nvarchar(max),
    @RequestGuid               uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The Site Collection identifier of the Site Collection which contains the specified Web Part. MUST NOT be NULL.

@DocID: The Document identifier (as specified in [MS-WSSFO2], section 2.2.1.2) of the Web Part Page where the Web Part is being added. MUST NOT be NULL.

@WebPartID: The Web Part identifier (as specified in [MS-WSSFO2], section 2.2.1.14) of the Web Part within the Site Collection. If the Web Part is successfully added, its Web Part type identifier property MUST be set to NULL and the following Web Part properties MUST be set using the passed-in values: Publishing Level, IsIncluded, Frame State, Web Part Zone, Zone Index, and Source. MUST NOT be NULL.

@Level: The Publishing Level for the Web Part. MUST NOT be NULL.

@IsIncluded: The Web Part Is Closed State for the Web Part. MUST NOT be NULL.

@FrameState: The Web Part Chrome State for the Web Part. MUST NOT be NULL.

@ZoneID: The Web Part Zone identifier of the Web Part Zone for the Web Part.

@PartOrder: The Web Part Zone Index for the Web Part.

@Source: The Web Part properties of the Web Part in either WPV2:WebPart format (as specified in [MS-WPPS], section 2.2.3.2) or HTML format.

@RequestGuid: The optional request identifier for the current request.

**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 completion.</td>
</tr>
<tr>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>1</td>
<td>Adding the Web Part failed</td>
</tr>
<tr>
<td>212</td>
<td>The specified Site Collection is Locked.</td>
</tr>
<tr>
<td>1816</td>
<td>The Quota for the specified Site Collection has been exceeded.</td>
</tr>
</tbody>
</table>

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

### 3.1.4.71 proc_RemoveSolution

The proc_RemoveSolution stored procedure is called to remove a sandboxed solution from the specified site collection. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_RemoveSolution (  
    @SiteId           uniqueidentifier,  
    @SolutionId       uniqueidentifier,  
    @Hash             nvarchar(50)  
);  
```

- **@SiteId**: The site collection identifier of the site collection from which the sandboxed solution is to be removed. This value MUST NOT be NULL.
- **@SolutionId**: The identifier of the sandboxed solution. This value MUST NOT be NULL.
- **@Hash**: The implementation-specific hash of the content of the sandboxed solution. This value MUST NOT be NULL.

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

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

### 3.1.4.72 proc_ResetSiteResourceUsageWarnings

The proc_ResetSiteResourceUsageWarnings stored procedure is called to do a bulk reset of the resource quota flags for this Site Collection. See 3.1.1.5 Quota Management for more information. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_ResetSiteResourceUsageWarnings ();  
```

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

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

### 3.1.4.73 proc_RestoreWebPartForDoc

The proc_ResoreWebPartForDoc stored procedure is called to restore Web Parts to a Web Part Page while restoring the Web Part Page to a previous version.

```sql
PROCEDURE proc_RestoreWebPartForDoc (  
    @SiteId            uniqueidentifier,  
    @DirName           nvarchar(256),  
    @LeafName          nvarchar(128),  
```
@Level             tinyint,
@OldVersion        int,
@RequestGuid       uniqueidentifier = null OUTPUT
);

@SiteId: The Site Collection identifier of the Site Collection containing the Web Part Page. MUST NOT be NULL.

@DirName: The Directory Name of the Web Part Page. MUST NOT be NULL.

@LeafName: The Leaf Name of the Web Part Page. MUST NOT be NULL.

@Level: The Publishing Level of the Web Part Page. MUST NOT be NULL.

@OldVersion: The version of the Web Part Page being restored. MUST NOT be NULL.

@RequestGuid: The optional request identifier for the current request.

Return 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 completion.</td>
</tr>
<tr>
<td>2</td>
<td>The Web Part Page was not found.</td>
</tr>
<tr>
<td>-2147467259</td>
<td>An error occurred while the stored procedure was running.</td>
</tr>
</tbody>
</table>

Result Sets: MUST NOT return any result sets.

3.1.4.74 proc_RevertInProgressWorkItem

The proc_RevertInProgressWorkItem stored procedure is called to revert the Work Item specified by the parameters. Reverting a Work Item means to mark as no longer In Progress Work Item and possibly perform exponential backoff on the Work Item Delivery Date; exponential backoff only occurs if the Work Item in question is marked for exponential backoff. Before any reverts occur, however, proc_RevertInProgressWorkItem deletes the indicated Work Item if it is both 10 or more days past its Delivery Date and marked for automatic deletion. The T-SQL syntax for the stored procedure is as follows:

PROCEDURE proc_RevertInProgressWorkItem(
    @ProcessingId            uniqueidentifier,
    @SiteId                  uniqueidentifier,
    @Id                      uniqueidentifier,
    @RequestGuid             uniqueidentifier = NULL OUTPUT
);

@ProcessingId: The Work Item Processing identifier of the Work Item Process. The server MUST only consider for deletion and revert a Work Item if it is associated with the Work Item Process specified by this parameter. MUST NOT be NULL.

@SiteId: The Site Collection identifier of the Site Collection. The server MUST only consider for deletion and revert a Work Item if it is associated with the Site Collection specified by the parameter. MUST NOT be NULL.
@Id: The Work Item identifier. The server MUST only revert a Work Item if it has the specified Work Item identifier. MUST NOT be NULL.

@RequestGuid: The optional request identifier for the current request.

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

Result Sets: MUST NOT return any result sets.

3.1.4.75 proc_RevertInProgressWorkItems

The proc_RevertInProgressWorkItems stored procedure is called to revert a set of Work Items specified by the parameters. Reverting a Work Item means to mark as no longer being In Progress Work Item and possibly perform exponential backoff on the Work Item Delivery Date; exponential backoff only occurs if the Work Item in question is marked for exponential backoff. Before any reverts occur, however, proc_RevertInProgressWorkItems deletes Work Items that are both 10 or more days past their Delivery Dates and marked for automatic deletion. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_RevertInProgressWorkItems(
    [ProcessingId] uniqueidentifier,
    [ProcessMachineId] uniqueidentifier,
    [SiteId] uniqueidentifier,
    [ParentId] uniqueidentifier,
    [WorkItemType] uniqueidentifier,
    [BatchId] uniqueidentifier,
    [AnyRemaining] int OUTPUT,
    [RequestGuid] uniqueidentifier = NULL OUTPUT
);
```

@ProcessingId: The Work Item Processing identifier of the Work Item Process. The server MUST only consider for deletion and revert a Work Item if it is associated with the Work Item Process specified by this parameter. MUST NOT be NULL.

@ProcessMachineId: This parameter MUST be ignored.

@SiteId: The Site Collection identifier of the Site Collection. If the parameter is not NULL, then the server MUST only consider for deletion and revert Work Items associated with this Site Collection. If the parameter is NULL, then the server MUST operate on the Work Items specified by the other parameters, regardless of associated Site Collection.

@ParentId: The Work Item Parent identifier of the Work Item. If the parameter is not NULL, then the server MUST only consider for deletion and revert Work Items which have this Work Item Parent identifier. If the parameter is NULL, then the server MUST operate on the Work Items specified by the other parameters, regardless of the value of their Work Item Parent identifier.

@WorkItemType: The Work Item type identifier of the Work Item type. The server MUST only consider for deletion and revert Work Items associated with this Work Item type. MUST NOT be NULL.

@BatchId: The Work Item Batch identifier of the Work Item Batch. If the parameter is not NULL, then the server MUST only consider for deletion and revert Work Items associated with this Work Item Batch. If the parameter is NULL, then the server MUST operate on Work Items specified by the other parameters, regardless of associated Work Item Batch.
@AnyRemaining: Specifies whether the stored procedure reverted any items. The protocol server MUST set this parameter to 1 if it reverted any Work Items. The server MUST set this parameter to 0 if it did not revert any Work Items.

@RequestGuid: The optional request identifier for the current request.

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

Result Sets: MUST NOT return any result sets.

3.1.4.76 proc_SetEventReceiverToSynchronous

The proc_SetEventReceiverToSynchronous stored procedure is called to set all registrations of the given event receiver to synchronous. The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_SetEventReceiverToSynchronous (
    @SiteId uniqueidentifier,
    @Assembly nvarchar(256),
    @Class nvarchar(256),
    @Type int
);
```

@SiteId: The site collection identifier of the site collection. This value MUST NOT be NULL.

@Assembly: The assembly name of the implementation of the event receiver.

@Class: The fully qualified class name of the implementation of the event receiver.

@Type: The type of the event receiver. @Type MUST be a value of Event Receiver type (as specified in [MS-WSSFO2], section 2.2.3.6).

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

Result Sets: MUST NOT return any result sets.

3.1.4.77 proc_TruncateResourceUsageDaily

The proc_TruncateResourceUsageDaily stored procedure is called to remove resource usage measurements from the daily solution resource usage log. The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_TruncateResourceUsageDaily (
    @IdStart bigint
);
```

@IdStart: The identifier of the resource usage measurements at which removal of records will begin. The server MUST remove all resource usage measurements from the daily solution resource usage log whose identifier is less than or equal to the specified value.

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

Result Sets: MUST NOT return any result sets.
3.1.4.78 proc_TruncateResourceUsageLog

The proc_TruncateResourceUsageLog stored procedure is called to remove resource usage measurements from the immediate solution resource usage log. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_TruncateResourceUsageLog (  @IdStart bigint  );
```

@IdStart: The identifier of resource usage measurements at which removal of records will begin. The server MUST remove all resource usage measurements from the immediate solution resource usage log whose identifier is less than or equal to the specified value.

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

Result Sets: MUST NOT return any result sets.

3.1.4.79 proc_TruncateResourceUsageWindowed

The proc_TruncateResourceUsageWindowed stored procedure is called to remove resource usage measurements from the windowed solution resource usage log. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_TruncateResourceUsageWindowed (  @IdStart bigint  );
```

@IdStart: The identifier of resource usage measurements at which removal of records will begin. The server MUST remove all resource usage measurements from the windowed solution resource usage log whose identifier is less than or equal to the specified value.

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

Result Sets: MUST NOT return any result sets.

3.1.4.80 proc_UpdateDataViewWhileSaving

The proc_UpdateDataViewWhileSaving stored procedure is called to create or update a Data View Web Part or data form Web Part. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_UpdateDataViewWhileSaving(  @SiteId                   uniqueidentifier,  @ListWebId                uniqueidentifier,  @ListId                   uniqueidentifier,  @ViewId                   uniqueidentifier,  @DisplayName              nvarchar(255),  @Type                     tinyint,  @Flags                    int,  @PageUrlID                 uniqueidentifier,  @Level                    tinyint,  @RequestGuid               uniqueidentifier = NULL OUTPUT  );
```
@SiteId: The Site Collection identifier for the Site Collection. MUST NOT be NULL.

@ListWebId: The Site identifier for the Site that contains the list. MUST NOT be NULL.

@ListId: The list identifier (as specified in [MS-WSSFO2], section 2.2.1.5) of the list. MUST NOT be NULL.

@ViewId: The GUID of the list View. MUST NOT be NULL.

@DisplayName: The Display Name for the Web Part. If this value is NULL the Web Part’s Display Name property MUST NOT be updated.

@Type: The Page type (as specified in [MS-WSSFO2], section 2.2.3.14) for the Web Part. If this value is NULL the Web Part’s Page type property MUST NOT be updated. If this value is Default View the View MUST be made the Default View.

@Flags: The View Flags (as specified in [MS-WSSFO2], section 2.2.2.12) for the View. If the VIEWFLAG_MOBILEDEFAULT (0x01000000) bit is set the View MUST be made the Default View for mobile devices.

@PageUrlID: The Document identifier (as specified in [MS-WSSFO2], section 2.2.1.2) of the Web Part Page containing the Web Part being updated. If this parameter is NULL the Document identifier (as specified in [MS-WSSFO2], section 2.2.1.2) of the Web Part Page MUST NOT be updated.

@Level: The Publishing Level of the Web Part. MUST NOT be NULL.

@RequestGuid: The optional request identifier for the current request.

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 completion.</td>
</tr>
<tr>
<td>1</td>
<td>The Data View Web Part was not successfully updated.</td>
</tr>
<tr>
<td>212</td>
<td>The specified Site Collection is Locked.</td>
</tr>
<tr>
<td>1816</td>
<td>The Quota for the specified Site Collection has been exceeded.</td>
</tr>
</tbody>
</table>

Result Sets: MUST NOT return any result sets.

3.1.4.81 proc_UpdateDocEventReceiver

The proc_UpdateDocEventReceiver stored procedure is called to update the registration of an event receiver for a specified document. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_UpdateDocEventReceiver(
    @DocUrl nvarchar(260),
    @Id uniqueidentifier,
    @Name nvarchar(256),
    @SiteId uniqueidentifier,
    @WebId uniqueidentifier,
    @ItemId int,
    @Synchronization int,
    @Type int,
    @SequenceNumber int,
)
```
@Assembly nvarchar(256),
@Class nvarchar(256),
@Data nvarchar(256),
@Filter nvarchar(256),
@Credential int,
@RequestGuid uniqueidentifier = NULL OUTPUT,
);

@DocUrl: The URL in store-relative form of the document.

@Id: The Event Receiver identifier (as specified in [MS-WSSFO2], section 2.2.1.3) of the event receiver. This value MUST NOT be NULL.

@Name: The name of the event receiver. This value MUST NOT be NULL.

@SiteId: The site collection identifier of the site collection which contains the document.

@WebId: The Site identifier (as specified in [MS-WSSFO2], section 2.2.1.10) of the site which contains the document.

@ItemId: Reserved. @ItemId MUST be 0.

@Synchronization: Specifies the synchronicity of the event receiver and the action triggering the event. The value MUST be an integer which is listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Default</td>
<td>For before event receivers, the server MUST run the event receiver synchronously. For after event receivers, the server is not required to run the event synchronously.</td>
</tr>
<tr>
<td>1</td>
<td>Synchronous</td>
<td>The server MUST run the event receiver using the same thread that is processing the request whose action triggered the event.</td>
</tr>
<tr>
<td>2</td>
<td>Asynchronous</td>
<td>The server MUST queue the task of running the event receiver. The server is not required to run the task using the same thread that is processing the request whose action triggered the event.</td>
</tr>
</tbody>
</table>

@Type: The type of the event receiver. @Type MUST be one of Event Receiver type (as specified in [MS-WSSFO2], section 2.2.3.6).

@SequenceNumber: The sequence number (1) of the event receiver. @SequenceNumber MUST be greater than or equal to zero and less than or equal to 65535.

@Assembly: The assembly name of the implementation of the event receiver. This value MUST NOT be NULL.

@Class: The fully qualified class name of the implementation of the event receiver. This value MUST NOT be NULL.

@Data: Additional data persisted on behalf of the event receiver implementation to be passed to the event receiver.

@Filter: Reserved. @Filter MUST be NULL.

@Credential: Reserved. @Credential MUST be zero.
@RequestGuid: The optional request identifier for the current request.

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>Update succeeded.</td>
</tr>
<tr>
<td>3</td>
<td>The document identified by @DocUrl is not found in the site identified by @WebId in the site collection identified by @SiteId.</td>
</tr>
<tr>
<td>87</td>
<td>Update failed.</td>
</tr>
</tbody>
</table>

Result Sets: MUST NOT return any result sets.

3.1.4.82 proc_UpdateEventReceiver

The proc_UpdateEventReceiver stored procedure is called to update the registration of a specified event receiver. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_UpdateEventReceiver(
    @Id                       uniqueidentifier,
    @Name                     nvarchar(256),
    @SiteId                   uniqueidentifier,
    @WebId                    uniqueidentifier,
    @HostId                   uniqueidentifier,
    @HostType                 int,
    @ItemId                   int,
    @DirName                  nvarchar(256),
    @LeafName                 nvarchar(128),
    @Synchronization         int,
    @Type                     int,
    @SequenceNumber          int,
    @Assembly                 nvarchar(256),
    @Class                    nvarchar(256),
    @Data                     nvarchar(256),
    @Filter                   nvarchar(256),
    @SourceId                 varbinary(512),
    @SourceType               int,
    @Credential               int,
    @ContextType              uniqueidentifier,
    @ContextEventType        uniqueidentifier,
    @ContextId                uniqueidentifier,
    @ContextObjectId         uniqueidentifier,
    @ContextCollectionId     uniqueidentifier,
    @RequestGuid              uniqueidentifier = NULL OUTPUT
);```

@Id: The Event Receiver identifier (as specified in [MS-WSSFO2], section 2.2.1.3) of the event receiver. This value MUST NOT be NULL.

@Name: The name of the event receiver. This value MUST NOT be NULL.

@SiteId: The Site Collection identifier of the site collection which contains the event host. This value MUST NOT be NULL.
@WebId: The Site identifier (as specified in [MS-WSSFO2], section 2.2.1.10) of the site which contains the event host. This value MUST NOT be NULL.

@HostId: The event host identifier of the event host of the event receiver. This value MUST NOT be NULL.

@HostType: The type of the event host of the event receiver. @HostType MUST be one of Event Host type (as specified in [MS-WSSFO2], section 2.2.3.5).

@ItemID: Reserved. @ItemID MUST be 0.

@DirName: Reserved. @DirName MUST be NULL.

@LeafName: Reserved. @LeafName MUST be NULL.

@Synchronization: Specifies the synchronicity of the event receiver and the action triggering the event. The value MUST be an integer which is listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Default</td>
<td>For before event receivers, the server MUST run the event receiver synchronously. For after event receivers, the server is not required to run the event synchronously.</td>
</tr>
<tr>
<td>1</td>
<td>Synchronous</td>
<td>The server MUST run the event receiver using the same thread that is processing the request whose action triggered the event.</td>
</tr>
<tr>
<td>2</td>
<td>Asynchronous</td>
<td>The server MUST queue the task of running the event receiver. The server is not required to run the task using the same thread that is processing the request whose action triggered the event.</td>
</tr>
</tbody>
</table>

@Type: The type of the event receiver. @Type MUST be one of the Event Receiver types specified in [MS-WSSFO2], section 2.2.3.6.

@SequenceNumber: The sequence number (1) of the event receiver. @SequenceNumber MUST be greater than or equal to zero and less than or equal to 65535.

@Assembly: The assembly name of the implementation of the event receiver. This value MUST NOT be NULL.

@Class: The fully qualified class name of the implementation of the event receiver. This value MUST NOT be NULL.

@Data: Additional data persisted on behalf of the event receiver implementation to be passed to the event receiver.

@Filter: Reserved. @Filter MUST be NULL.

@SourceId: The event receiver source identifier of the event receiver. This is the Feature identifier (as specified in [MS-WSSFO2], section 2.2.1.4) of the feature if the event receiver is added via a feature. This is the Content type identifier (as specified in [MS-WSSFO2], section 2.2.1.1) of the content type if the event receiver is added via a content type. Otherwise the event receiver source identifier MUST be NULL.

@SourceType: The Event Receiver Source type of the event receiver. @SourceType MUST be one of Event Receiver Source Type values.

@Credential: Reserved. @Credential MUST be 0.
@ContextType: The context type identifier of the event receiver.

@ContextEventType: Reserved. @ContextEventType MUST be NULL.

@ContextId: The context identifier of the event receiver.

@ContextObjectId: The context object identifier for the Event Host of the event receiver.

@ContextCollectionId: The context collection identifier of the event receiver.

@RequestGuid: The optional request identifier for the current request.

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>Update succeeded.</td>
</tr>
<tr>
<td>87</td>
<td>Update failed because the specified event receiver does not exist in the specified site collection or the site collection does not exist or the Event Receiver type of the event receiver is identified by 32767 (as specified in [MS-WSSFO2], section 2.2.3.6).</td>
</tr>
</tbody>
</table>

Result Sets: MUST NOT return any result sets.

### 3.1.4.83 proc_UpdateListFormWhileSaving

The proc_UpdateListFormWhileSaving stored procedure is called to create or update a List Form Web Part. The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_UpdateListFormWhileSaving(
    @SiteId                   uniqueidentifier,
    @ListWebId                uniqueidentifier,
    @ListId                   uniqueidentifier,
    @ViewId                   uniqueidentifier,
    @Flags                    int,
    @Type                     tinyint,
    @PageUrlID                uniqueidentifier,
    @Level                    tinyint,
    @RequestGuid              uniqueidentifier
);
```

@SiteId: The Site Collection identifier for the Site Collection. MUST NOT be NULL.

@ListWebId: The Site identifier for the Site that contains the list. MUST NOT be NULL.

@ListId: The list identifier (as specified in [MS-WSSFO2], section 2.2.1.5) of the list. MUST NOT be NULL.

@ViewId: The GUID of the list View. MUST NOT be NULL.

@Flags: The View Flags (as specified in [MS-WSSFO2], section 2.2.2.12) for the View. If @Flags is NULL the list Form View Flags (as specified in [MS-WSSFO2], section 2.2.2.12) are not updated.

@Type: The Page type (as specified in [MS-WSSFO2], section 2.2.3.12) for the list Form Web Part. If this value is NULL the list Form Web Part's Page type property MUST NOT be updated.
@PageUrlID: The Document identifier (as specified in [MS-WSSFO2], section 2.2.1.2) of the Web Part Page containing the Web Part being updated. If this parameter is NULL the Document identifier (as specified in [MS-WSSFO2], section 2.2.1.2) of the list Form Web Part MUST not be updated.

@Level: Publishing Level. MUST NOT be NULL.

@RequestGuid: The optional request identifier for the current request.

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 completion.</td>
</tr>
<tr>
<td>1</td>
<td>The list Form was not successfully Updated.</td>
</tr>
<tr>
<td>212</td>
<td>The specified Site Collection is Locked.</td>
</tr>
<tr>
<td>1816</td>
<td>The Quota for the specified Site Collection has been exceeded.</td>
</tr>
</tbody>
</table>

Result Sets: MUST NOT return any result sets.

3.1.4.84 proc_UpdateListItemWorkflowInstanceData

The proc_UpdateListItemWorkflowInstanceData stored procedure is called to update a workflow. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_UpdateListItemWorkflowInstanceData(
    @SiteId                   uniqueidentifier,
    @WebId                    uniqueidentifier,
    @ListId                   uniqueidentifier,
    @ItemId                   int,
    @WorkflowInstanceId       uniqueidentifier,
    @InstanceData             varbinary(max),
    @InstanceDataSize         int,
    @Modifications            nvarchar(max),
    @WakeupTime               datetime,
    @InstanceDataVersionId    int,
    @Status1                  int,
    @Status2                  int,
    @Status3                  int,
    @Status4                  int,
    @Status5                  int,
    @Status6                  int,
    @Status7                  int,
    @Status8                  int,
    @Status9                  int,
    @Status10                 int,
    @ActivityDetails          varbinary(max),
    @WorkflowCompleted        bit,
    @WorkflowSuspended        bit,
    @WorkflowFaulting         bit,
    @WorkflowTerminated       bit,
    @WorkflowCanceled         bit,
    @UnlockInstance           bit,
    @ProcessingId             uniqueidentifier,
    @InternalState            int OUTPUT,
    @RequestGuid              uniqueidentifier = NULL OUTPUT
)
```
@SiteId: The Site Collection identifier of the Site Collection which contains the Workflow. The protocol server MUST update the Site Collection Quota (section 3.1.1.5) to reflect the change in space used by the Workflow.

@WebId: The Site identifier (as specified in [MS-WSSFO2], section 2.2.1.10) of the Site which contains the Workflow.

@ListId: The list identifier of the list which contains the list Item the Workflow was created for.

@ItemId: The list Item identifier (as specified in [MS-WSSFO2], section 2.2.1.6) of the list Item for which the Workflow was created.

@WorkflowInstanceId: The Workflow identifier of the Workflow. The server MUST NOT update the Workflow if it is a completed workflow. The server MUST set the modification date and time of the workflow to the date and time in UTC when the procedure was called.

@InstanceData: The workflow instance data of the Workflow.

@InstanceDataSize: The size of @InstanceData. If @InstanceData is NULL, @InstanceDataSize MUST contain the value 0.

@Modifications: The Workflow Modifications of the Workflow. If this value is an empty string, or if any of @WorkflowCompleted, @WorkflowCanceled or @WorkflowTerminated contain the value 1, the server MUST NOT update the Workflow Modifications field of the Workflow.

@WakeupTime: The date and time in UTC for the server to wake the Workflow to resume processing. If @WakeupTime is not NULL, @UnlockInstance contains the value 1, and all of @WorkflowCompleted, @WorkflowCanceled and @WorkflowTerminated contain the value 0, the server MUST create a Work Item to wake up the Workflow.

@InstanceDataVersionId: MUST contain the value 0.

@Status1: The Workflow Status1 value for the Workflow. If this value is NOT NULL, the server MUST set the Workflow Status1 field of the Workflow to this value. Otherwise, the server MUST update the Workflow Status1 value as follows:

- If @WorkflowFaulting contains the value 1, the server MUST update the Workflow Status1 value to WFSTAT_FAULTING_RETRY.
- If @WorkflowTerminated contains the value 1 and @WorkflowCompleted contains the value 0, the server MUST update the Workflow Status1 value to WFSTAT_FAULTING.
- If @WorkflowCompleted contains the value 1, the server MUST update the Workflow Status1 value to WFSTAT_COMPLETED.
- If the current Workflow Status1 value is WFSTAT_FAULTING, the server MUST update the Status1 value to WFSTAT_INPROGRESS.
- In other cases, the server MUST NOT update the Workflow Status1 value.

@Status2: This parameter MUST be ignored.

@Status3: This parameter MUST be ignored.

@Status4: This parameter MUST be ignored.
@Status5: This parameter MUST be ignored.

@Status6: This parameter MUST be ignored.

@Status7: This parameter MUST be ignored.

@Status8: This parameter MUST be ignored.

@Status9: This parameter MUST be ignored.

@Status10: This parameter MUST be ignored.

@ActivityDetails: An implementation defined binary payload containing the activity details.

@WorkflowCompleted: Determines whether the Workflow is marked as completed. Once a Workflow is marked as completed, it cannot be marked as not completed. This value MUST NOT be NULL. When @WorkflowCompleted contains the value 1, the server MUST update workflow internal state of the Workflow to add the WFS_COMPLETED flag (0x0004) and remove the WFS_RUNNING (0x0002) and WFS_HASNEWEVENTS New Events (0x0400) flags.

@WorkflowSuspended: Determines whether the Workflow is marked as suspended. This value MUST NOT be NULL. When @WorkflowSuspended contains the value 1, the server MUST update workflow internal state of the Workflow to add the WFS_SUSPENDED (0x0100) flag.

@WorkflowFaulting: Determines whether the Workflow is marked as faulting. This value MUST NOT be NULL. When @WorkflowFaulting contains the value 1, the server MUST update workflow internal state of the Workflow to add the WFS_Faulting flag.

@WorkflowTerminated: Determines whether the Workflow is marked as terminated. Once a Workflow is marked as terminated, it cannot be marked as not terminated. This value MUST NOT be NULL. When @WorkflowTerminated contains the value 1, the server MUST update workflow internal state of the Workflow to add the WFS_Terminated flag.

@WorkflowCanceled: Determines whether the Workflow is marked as canceled. Once a Workflow is marked as canceled, it cannot be marked as not canceled. This value MUST NOT be NULL. When @WorkflowCanceled contains the value 1, the server MUST update workflow internal state of the Workflow to add the WFS_Cancelled flag and remove the WFS_Running and WFS_HASNEWEVENTS flags. If any of @WorkflowCompleted, @WorkflowCanceled or @WorkflowTerminated contain the value 1, the server MUST set the workflow instance data for the workflow to null and the instance data size to 0.

@UnlockInstance: Determines whether the Workflow is unlocked. This value MUST NOT be NULL. When @UnlockInstance contains the value 1, or if any of @WorkflowCompleted, @WorkflowCanceled or @WorkflowTerminated contain the value 1, the server MUST update workflow internal state of the Workflow to remove the WFS_Locked flag.

@ProcessingId: The workflow processing identifier of the workflow process running the Workflow.

@InternalState: The server MUST ignore the input value of this parameter. The server MUST set the output value of this parameter to the workflow internal state of the Workflow after the procedure action is complete.

@RequestGuid: The optional request identifier for the current request.

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 completion.</td>
</tr>
<tr>
<td>5</td>
<td>Access is denied.</td>
</tr>
<tr>
<td>82</td>
<td>Failed to update the workflow.</td>
</tr>
</tbody>
</table>

**Result Sets:** MUST return one empty result set.

### 3.1.4.85 proc_UpdateListItemWorkflowLock

The `proc_UpdateListItemWorkflowLock` stored procedure is called to update and lock or unlock a workflow. The T-SQL syntax for the stored procedure is as follows:

```tsql
PROCEDURE proc_UpdateListItemWorkflowLock(
    @SiteId                  uniqueidentifier,
    @WorkflowInstanceId      uniqueidentifier,
    @Lock                    bit,
    @ProcessingId            uniqueidentifier,
    @EventsNotDelivered      bit = 0,
    @NewStatus               int = -1,
    @RequestGuid             uniqueidentifier = NULL OUTPUT
)
```

**@SiteId:** The Site Collection identifier of the Site Collection containing the Workflow. The server MUST update the Site Collection Quota (section 3.1.1.5) to reflect the change in space used by the Workflow.

**@WorkflowInstanceId:** The Workflow identifier of the Workflow to be updated.

**@Lock:** Determines whether the Workflow will be locked or unlocked. This value MUST be either 0 or 1. When set to 1, the server MUST lock the Workflow. The server MUST add the Locked flag (0x0001) to the workflow internal state of the Workflow. When set to 0, the server MUST unlock the Workflow. The server MUST remove the Locked flag (0x0001) from the workflow internal state of the Workflow, and if the workflow internal state contains the Not Started flag (0x0800), the server MUST remove the Not Started flag and add the Running flag (0x0002).

**@ProcessingId:** The workflow processing identifier of the workflow process running the Workflow. If @Lock contains the value 0, the server MUST ignore the value in @ProcessingId.

**@EventsNotDelivered:** Indicates whether the completed or terminated Workflow has outstanding events. This value MUST NOT be NULL. If @Lock is set to 1, this value MUST be set to 0.

**@NewStatus:** The optional Workflow Status value for the Workflow to be updated.

**@RequestGuid:** The optional request identifier for the current request.

**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 completion.</td>
</tr>
<tr>
<td>5</td>
<td>Error: Access denied.</td>
</tr>
<tr>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>82</td>
<td>Error: Failed to update or lock the workflow.</td>
</tr>
</tbody>
</table>

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

### 3.1.4.86 proc_UpdateListViewFormWebPartSource

The `proc_UpdateListViewFormWebPartSource` stored procedure is called to update the `Source` property of an existing Web Part. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_UpdateListViewFormWebPartSource(
    @SiteId uniqueidentifier,
    @WebPartId uniqueidentifier,
    @Source nvarchar(max),
    @RequestGuid uniqueidentifier = NULL OUTPUT
);
```

- **@SiteId:** The Site Collection identifier of the Site Collection which contains the specified Web Part. MUST NOT be NULL.
- **@WebPartId:** The Web Part identifier (as specified in [MS-WSSFO2], section 2.2.14) of the Web Part within the Site Collection. MUST NOT be NULL.
- **@Source:** The Web Part properties of the Web Part in either WPV2:WebPart format (as specified in [MS-WPPS], section 2.2.3.2) or HTML format.
- **@RequestGuid:** The optional request identifier for the current request.

**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 completion.</td>
</tr>
<tr>
<td>1</td>
<td>Updating the Web Part Source property failed</td>
</tr>
<tr>
<td>13</td>
<td>Web Part for the given @SiteId and @WebPartId does not exist.</td>
</tr>
<tr>
<td>212</td>
<td>The specified Site Collection is Locked.</td>
</tr>
<tr>
<td>1816</td>
<td>The Quota for the specified Site Collection has been exceeded.</td>
</tr>
</tbody>
</table>

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

### 3.1.4.87 proc_UpdateListViewToDataViewForSite

The `proc_UpdateListViewToDataViewForSite` is called to change the Web Part Type of a set of List Web Parts to a different Web Part Type.

```sql
PROCEDURE proc_UpdateListViewToDataViewForSite (    
    @DataViewId uniqueidentifier, 
    @FeatureId uniqueidentifier, 
    @ScopeId uniqueidentifier, 
    @ListViewId uniqueidentifier
);
```
@ DataViewId: The Web Part type identifier of the new Web Part Type. The server MUST NOT update the Web Part if the web part has CAML (that is, tp_View is not NULL), if the View Flags (as specified in [MS-WSSFO2], section 2.2.2.12) do not contain the value 1 (HTML view), or if the View Flags contain any bits in the mask 0x040A0810.

@ FeatureId: The identifier of the feature that defines the Web Parts.

@ ScopeId: The Site Collection identifier of the Site Collection containing the web parts.

@ ListViewId: The Web Part type identifier of the web parts to update.

Return values: An integer which the protocol client MUST ignore.

Result Sets: MUST NOT return any result sets.

3.1.4.88 proc_UpdateListViewToDataViewForWeb

The proc_UpdateListViewToDataViewForWeb is called to change the Web Part Type of a set of List Web Parts to a different Web Part Type.

PROCEDURE proc_UpdateListViewToDataViewForWeb (  
@DataViewId          uniqueidentifier,  
@FeatureId           uniqueidentifier,  
@ScopeId             uniqueidentifier,  
@ListViewId          uniqueidentifier  
);  

@ DataViewId: The Web Part type identifier of the new Web Part Type. The server MUST NOT update the Web Part if the web part has CAML (that is, tp_View is not NULL), if the View Flags (as specified in [MS-WSSFO2], section 2.2.2.12) do not contain the value 1 (HTML view), or if the View Flags contain any bits in the mask 0x040A0810.

@ FeatureId: The identifier of the feature that defines the Web Parts.

@ ScopeId: The Site identifier for the Site that contains the web parts.

@ ListViewId: The Web Part type identifier of the Web Parts to update.

Return values: An integer which the protocol client MUST ignore.

Result Sets: MUST NOT return any result sets.

3.1.4.89 proc_UpdateSiteResourceUsage

The proc_UpdateSiteResourceUsage stored procedure is called to update resource usage values for a site collection. The T-SQL syntax for the stored procedure is as follows:

PROCEDURE proc_UpdateSiteResourceUsage (  
@SiteId              uniqueidentifier,  
@CurrentResourceUsage float,  
@AverageResourceUsage float  
);
@SiteId: The site collection identifier of the site collection for which resource usage values are being updated.

@CurrentResourceUsage: The resource usage value for the current monitoring interval.

@AverageResourceUsage: The mean resource usage value for the specified site collection over the configured retention interval for resource usage data.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.1.4.90 proc_UpdateSolution

The proc_UpdateSolution stored procedure is called to update information about a sandboxed solution. The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_UpdateSolution (  
    @SiteId uniqueidentifier,  
    @SolutionId uniqueidentifier,  
    @Hash nvarchar(50),  
    @ValidatorsHash char(64),  
    @ValidationErrorUrl nvarchar(1024),  
    @ValidationErrorMessage nvarchar(1024)  
);  
```

@SiteId: The site collection identifier of the site collection for the sandboxed solution. This value MUST NOT be NULL.

@SolutionId: The identifier of the sandboxed solution. This value MUST NOT be NULL.

@Hash: The implementation-specific hash of the content of the sandboxed solution. This value MUST NOT be NULL.

@ValidatorsHash: The implementation-specific hash of the validators that validated the sandboxed solution. This value MUST NOT be NULL.

@ValidationErrorUrl: If validation of the sandboxed solution failed, MUST contain the URL with more information about the validation failure.

@ValidationErrorMessage: If validation of the sandboxed solution failed, MUST contain the specific error message of the validation failure.

Return Code Values:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful execution.</td>
</tr>
<tr>
<td>1</td>
<td>The sandboxed solution cannot be found.</td>
</tr>
</tbody>
</table>

Result Sets: MUST NOT return any result sets.
3.1.4.91 proc_UpdateSolutionResourceUsage

The proc_UpdateSolutionResourceUsage stored procedure is called to update resource usage values for a sandboxed solution. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_UpdateSolutionResourceUsage (  
    @SiteId uniqueidentifier,  
    @SolutionId uniqueidentifier,  
    @ResourceQuota float,  
    @RecentInvocations int
);  
```

@SiteId: The site collection identifier of the site collection containing the sandboxed solution.

@SolutionId: The identifier of the sandboxed solution.

@ResourceQuota: The resource usage value for the sandboxed solution over the current monitoring interval.

@RecentInvocations: The count of invocations of code from this sandboxed solution over the current monitoring interval.

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

Result Sets: MUST NOT return any result sets.

3.1.4.92 proc_UpdateViewWhileSaving

The proc_UpdateViewWhileSaving stored procedure is called to create or update a list View. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_UpdateViewWhileSaving (  
    @SiteId uniqueidentifier,  
    @ListWebId uniqueidentifier,  
    @ListId uniqueidentifier,  
    @ViewId uniqueidentifier,  
    @View tCompressedString,  
    @DisplayName nvarchar(255),  
    @ContentTypeId varbinary(512),  
    @Type tinyint,  
    @Flags int,  
    @BaseViewID tinyint,  
    @PageUrlID uniqueidentifier,  
    @Level tinyint,  
    @RequestGuid uniqueidentifier = NULL OUTPUT
);  
```

@SiteId: The Site Collection identifier of the Site Collection. MUST NOT be NULL.

@ListWebId: The Site identifier for the Site that contains the list. MUST NOT be NULL.

@ListId: The list identifier (as specified in [MS-WSSFO2], section 2.2.1.5) of the list. MUST NOT be NULL.

@ViewId: The GUID of the list View. MUST NOT be NULL.
@View: CAML XML of the View.

@DisplayName: The Display Name of the list View. If @DisplayName is NULL the Display Name property MUST NOT be updated.

@ContentTypeID: The Content type identifier (as specified in [MS-WSSFO2], section 2.2.1.1) of the list Items in the list to be displayed in the list View. If @ContentTypeID is NULL the Content type identifier property MUST NOT be updated.

@Type: The Page type (as specified in [MS-WSSFO2], section 2.2.3.12) of the list View. If @Type is NULL the Page type property MUST NOT be updated. If @Type has a value of Default View the View MUST be made the Default View for the list.

@Flags: This field is a bitmask, as specified in View Flags (as specified in [MS-WSSFO2], section 2.2.2.12) of the list View. When this property contains NULL, the View Flags (as specified in [MS-WSSFO2], section 2.2.2.12) property MUST NOT be updated. Otherwise, the protocol server MUST update the view flags property to the value of @Flags, and the default list view MUST be set depending on the bit values that are specified in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VIEWFLAG_MOBILEDEFAULT (0x01000000) bit set</td>
<td>The View for Mobile Devices.</td>
</tr>
<tr>
<td>VIEWFLAG_CONTENTTYPEDEFAULT (0x10000000) bit set</td>
<td>If the folders match the Content type identifier (as specified in [MS-WSSFO2], section 2.2.1.1) and the view of the folder is either not selected or not valid then use the list View</td>
</tr>
</tbody>
</table>

@BaseViewID: The base view identifier of the list View. If @BaseViewID is NULL the base view identifier property MUST NOT be updated.

@PageUrlID: The Document identifier (as specified in [MS-WSSFO2], section 2.2.1.2) of the Web Part Page. If this parameter is NULL the Document identifier (as specified in [MS-WSSFO2], section 2.2.1.2) of the Web Part Page MUST NOT be updated.

@Level: Publishing Level of the list View. MUST NOT be NULL.

@RequestGuid: The optional request identifier for the current request.

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 completion.</td>
</tr>
<tr>
<td>1</td>
<td>View was not successfully Updated.</td>
</tr>
<tr>
<td>212</td>
<td>The specified Site Collection is Locked.</td>
</tr>
<tr>
<td>1359</td>
<td>An internal error occurred.</td>
</tr>
<tr>
<td>1816</td>
<td>The Quota for the specified Site Collection has been exceeded.</td>
</tr>
</tbody>
</table>

Result Sets: MUST NOT return any result sets.
3.1.4.93 proc_UpdateWebPart

The proc_UpdateWebPart stored procedure is called to update the state of an existing Web Part. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_UpdateWebPart(
    @SiteId uniqueidentifier,
    @DirName nvarchar(256),
    @LeafName nvarchar(128),
    @Level tinyint OUT PUT,
    @bAllUser bit,
    @SystemID varbinary(512),
    @WebPartID uniqueidentifier,
    @WebPartTypeID uniqueidentifier,
    @Assembly nvarchar(255),
    @Class nvarchar(255),
    @SolutionId uniqueidentifier,
    @bCheckLock bit,
    @IsIncluded bit,
    @FrameState tinyint,
    @ZoneID nvarchar(64),
    @PartOrder int,
    @TheFlags int,
    @TheType tinyint,
    @TheBaseViewID tinyint,
    @AllUsersProperties varbinary(max),
    @PerUserProperties varbinary(max),
    @WebPartIdProperty nvarchar(255),
    @RequestGuid uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The Site Collection identifier of the site collection which contains the Web Part.

@DirName: The Directory Name of the Web Part Page that contains the Web Part.

@LeafName: The Leaf Name of the Web Part Page that contains the Web Part.

@Level: The Publishing Level of the Web Part Page. The value is returned as an output parameter and MUST be the same value passed in or Draft. The value is changed to Draft if the Web Part Page is in a Document Library, @Level is Published, @bCheckLock is 1, @bAllUser is 1, @SystemID references an existing user in the Site Collection, the Web Part Page is Moderated or has minor version control enabled, and creation of a new version of the Web Part Page succeeded.

@bAllUser: Specifies whether to update the Web Part for the Shared View or personal View of the Web Part Page. If this flag is set to 1 the Web Part is updated for the Shared View of the Web Part Page and the changes are available to All Users. If this flag is set to 0 @SystemID is used to update the Web Part for the current users personal View of the Web Part Page and is available only to the current user.

@SystemID: The SystemID of the current user. If the Web Part Page is moderated or has minor version control enabled then @SystemID is used to track who is modifying the Web Part.

@WebPartID: The Web Part identifier (as specified in [MS-WSSFO2], section 2.2.1.14) of the Web Part. MUST NOT be NULL.

@WebPartTypeID: The Web Part type identifier of the Web Part being updated. MUST NOT be NULL.
@Assembly: The fully qualified name of the assembly that implements the web part.

@Class: The name of the .NET class that implements the web part.

@SolutionId: The identifier of the sandboxed solution that implements the web part.

@bCheckLock: If this flag is set to 1, check if the document is in a state where it can be modified, if it cannot be modified, return specific Return Code values, defined in the following Return Code Values table, that explain why it cannot be modified. If this flag is set to 0, the checks made when this flag is set to 1, are bypassed.

@IsIncluded: The Web Part Is Closed State of the Web Part.

@FrameState: The Web Part Chrome State of the Web Part.

@ZoneID: The name of the Web Part Zone identifier of the Web Part Zone that contains the Web Part.

@PartOrder: The Web Part Zone Index of the Web Part.

@TheFlags: The View Flags, as specified in [MS-WSSFO2], section 2.2.12, of the Web Part.

@TheType: The page type of the web part page containing the web part.

@TheBaseViewID: The base view identifier of the web part.

@AllUsersProperties: A binary payload containing 0 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.

@PerUserProperties: A binary payload containing 0 or more personalizable properties of the Web Part. If this value is NULL then default values will be used for all of the personalizable properties of the Web Part.

@WebPartIdProperty: The HTML (HyperText Markup Language) ID attribute of the Web Part. May be NULL. If not NULL, it MUST be unique per Web Part Page.

@RequestGuid: The optional request identifier for the current request.

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 completion.</td>
</tr>
<tr>
<td>-2147467259</td>
<td>An error occurred while the stored procedure was running.</td>
</tr>
<tr>
<td>2</td>
<td>The Web Part Page cannot be found or @SiteId, @DirName or @LeafName is NULL.</td>
</tr>
<tr>
<td>3</td>
<td>The Web Part Page is Moderated or has minor version control enabled, and a new version of the Web Part Page cannot be created.</td>
</tr>
<tr>
<td>5</td>
<td>The Web Part being updated is not on the Web Part Page.</td>
</tr>
<tr>
<td>12</td>
<td>@bCheckLock is 1, @bAllUser is 0 and @Level is Checked Out.</td>
</tr>
<tr>
<td>33</td>
<td>@bCheckLock is 1, @bAllUser is 1, and the Web Part Page is not the Current Version.</td>
</tr>
<tr>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>87</td>
<td>The Web Part Page is in a Document Library, @Level is Published, @bCheckLock is 1, @bAllUser is 1, @SystemID references an existing user in the Site Collection, the Web Part Page is Moderated or has minor version control enabled, and a new Draft version of the Web Part Page cannot be created.</td>
</tr>
<tr>
<td>158</td>
<td>@bCheckLock is 1, @bAllUser is 1, @Level is NOT Checked Out and the Web Part Page is required to be Checked Out before it is modified.</td>
</tr>
<tr>
<td>212</td>
<td>The Site Collection is Locked.</td>
</tr>
<tr>
<td>1816</td>
<td>The Quota for the Site Collection has been exceeded.</td>
</tr>
</tbody>
</table>

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

### 3.1.4.94 proc_UpdateWebPartCache

The proc_UpdateWebPartCache stored procedure is called to write the private data cache of the specified Web Part to the database. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_UpdateWebPartCache(
    @SiteId                   uniqueidentifier,
    @DirName                  nvarchar(256),
    @LeafName                 nvarchar(128),
    @Level                    tinyint OUTPUT,
    @bAllUser                 bit,
    @SystemID                 varbinary(512),
    @WebPartID                uniqueidentifier,
    @Cache                    varbinary(max),
    @RequestGuid              uniqueidentifier = NULL OUTPUT
);
```

**@SiteId:** The Site Collection identifier of the Site Collection which contains the specified Web Part.

**@DirName:** The Directory Name of the Web Part Page containing the requested Web Part.

**@LeafName:** The Leaf Name of the Web Part Page containing the requested Web Part.

**@Level:** The Publishing Level of the Web Part Page. The value is returned as an output parameter and MUST be the same value as passed into the procedure.

**@bAllUser:** A bit flag specifying whether to update Web Part cache for All Users or just the current user. If this flag is set to 0, proc_UpdateWebPartCache MUST update Web Part cache just for the current user specified by @SystemID. If this flag is set to 1, proc_UpdateWebPartCache MUST update Web Part cache for All Users.

**@SystemID:** The SystemID of the current user.

**@WebPartID:** The Web Part identifier (as specified in [MS-WSSFO2] section 2.2.1.15) of the Web Part.

**@Cache:** The private data cache of the Web Part.

**@RequestGuid:** The optional request identifier for the current request.

**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 completion.</td>
</tr>
<tr>
<td>2</td>
<td>The specified Web Part Page cannot be found.</td>
</tr>
<tr>
<td>5</td>
<td>The Web Part as specified by @WebPartID exists on a different Web Part Page within the Site Collection.</td>
</tr>
<tr>
<td>212</td>
<td>The Site Collection is Locked.</td>
</tr>
<tr>
<td>1816</td>
<td>The Quota for the specified Site Collection has been exceeded.</td>
</tr>
<tr>
<td>-2147467259</td>
<td>An error occurred while the stored procedure was running.</td>
</tr>
</tbody>
</table>

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

### 3.1.4.95 proc_UpdateWebPartIsIncluded

The proc_UpdateWebPartIsIncluded stored procedure is called to customize or personalize four specific properties of a Web Part: its Web Part Is Closed State, which Web Part Zone it is in, its Web Part Zone Index, and its Web Part Chrome State. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_UpdateWebPartIsIncluded(
    @SiteId uniqueidentifier,
    @DirName nvarchar(256),
    @LeafName nvarchar(128),
    @Level tinyint OUTPUT,
    @bAlluser bit,
    @UserID int,
    @WebPartID uniqueidentifier,
    @bCheckLock bit,
    @IsIncluded bit,
    @ZoneID nvarchar(64),
    @PartOrder int,
    @FrameState tinyint,
    @RequestGuid uniqueidentifier = NULL OUTPUT
);
```

**@SiteId:** The Site Collection identifier of the site collection which contains the Web Part.

**@DirName:** The Directory Name of the Web Part Page that contains the Web Part.

**@LeafName:** The Leaf Name of the Web Part Page that contains the Web Part.

**@Level:** The Publishing Level of the Web Part Page for the current user. The value is returned as an output parameter and MUST be the same as the input value or Draft. The value is changed to Draft if the Web Part Page is in a Document Library, @Level is Published, @bCheckLock is 1, @bAlluser is 1, @UserID references an existing user in the Site Collection, the Web Part Page is Moderated or has minor version control enabled, and creation of a new version of the Web Part Page succeeded.

**@bAlluser:** A bit flag specifying whether to update the Web Part for the Shared View or personal View of the Web Part Page. If this flag is set to 1, the Web Part is updated for the Shared View of the Web Part Page and the changes are available to All Users. If this flag is set to 0, the Web Part is updated for the current user's personal View of the Web Part Page.
@UserID: The User identifier (as specified in [MS-WSSFO2], section 2.2.1.12) of the current user.

@WebPartID: The Web Part identifier (as specified in [MS-WSSFO2], section 2.2.1.14) of the Web Part. This MUST NOT be NULL.

@bCheckLock: If this flag is set to 1, check if the document is in a state where it can be modified, if it cannot be modified, return specific Return Code values, defined in the following Return Code Values table, that explain why it cannot be modified. If this flag is set to 0, the checks made when this flag is set to 1 are bypassed.

@IsIncluded: The Web Part Is Closed State of the Web Part.

@ZoneID: The Web Part Zone identifier of the Web Part Zone in which to put the Web Part.

@PartOrder: The Web Part Zone Index for the Web Part.

@FrameState: The Web Part Chrome State of the Web Part.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: The stored procedure MUST return an integer listed in the following table:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful completion.</td>
</tr>
<tr>
<td>2</td>
<td>The Web Part Page cannot be found or @SiteId, @DirName or @LeafName is NULL.</td>
</tr>
<tr>
<td>3</td>
<td>The Web Part Page is Moderated or has minor version control enabled, and a new version of the Web Part Page cannot be created.</td>
</tr>
<tr>
<td>5</td>
<td>The Web Part is not on the Web Part Page.</td>
</tr>
<tr>
<td>12</td>
<td>@bCheckLock is 1, @bAllUser is 0 and @Level is Checked Out.</td>
</tr>
<tr>
<td>33</td>
<td>@bCheckLock is 1, @bAllUser is 1, and the Web Part Page is not the Current Version.</td>
</tr>
<tr>
<td>87</td>
<td>The Web Part Page is in a Document Library, @Level is Published, @bCheckLock is 1, @bAllUser is 1, @UserId references an existing user in the Site Collection, the Web Part Page is Moderated or has minor version control enabled, and a new Draft version of the Web Part Page cannot be created.</td>
</tr>
<tr>
<td>160</td>
<td>The Web Part Page is in a Document Library, @Level is Published, @bCheckLock is 1, @bAllUser is 1, the Web Part Page is Moderated or has minor version control enabled, and @UserId is NULL.</td>
</tr>
<tr>
<td>212</td>
<td>The Site Collection is Locked.</td>
</tr>
<tr>
<td>1816</td>
<td>The Quota for the Site Collection has been exceeded.</td>
</tr>
<tr>
<td>-2147467259</td>
<td>An error occurred while the stored procedure was running.</td>
</tr>
</tbody>
</table>

Result Sets: MUST NOT return any result sets.

3.1.4.96 proc_UpdateWebPartProps

The proc_UpdateWebPartProps stored procedure is called to update the properties of an existing Web Part. The T-SQL syntax for the stored procedure is as follows:
PROCEDURE proc_UpdateWebPartProps(
    @SiteId uniqueidentifier,
    @WebPartID uniqueidentifier,
    @Type tinyint,
    @Flags int,
    @IsIncluded bit,
    @FrameState tinyint,
    @AllUsersproperties varbinary(max),
    @PerUserproperties varbinary(max),
    @WebPartIdProperty nvarchar(255),
    @Level tinyint OUTPUT,
    @RequestGuid uniqueidentifier = NULL OUTPUT
);

@SiteId: The Site Collection identifier of the site collection which contains the Web Part.

@WebPartID: The Web Part identifier (as specified in [MS-WSSFO2], section 2.2.1.14) of the Web Part. This value MUST NOT be NULL.

@Type: The Page Type, as specified in [MS-WSSFO2], section 2.2.3.14, of the Web Part Page that contains the Web Part.

@Flags: The View Flags, as specified in [MS-WSSFO2], section 2.2.2.12, of the Web Part.

@IsIncluded: The Web Part Is Closed State of the Web Part.

@FrameState: The Web Part Chrome State of the Web Part.

@AllUsersProperties: A serialized representation of 0 or more customizable properties of the Web Part. If this value is NULL then default values will be used for all of the Customizable properties of the Web Part.

@PerUserProperties: A serialized representation of 0 or more personalizable properties of the Web Part. If this value is NULL then default values will be used for all of the personalizable properties of the Web Part.

@WebPartIdProperty: The HTML (HyperText Markup Language) ID attribute of the Web Part. May be NULL. If not NULL, it MUST be unique per Web Part Page.

@Level: The Publishing Level of the Web Part Page containing the Web Part. The value is returned as an output parameter and MUST be the same value passed in.

@RequestGuid: The optional request identifier for the current request.

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 completion.</td>
</tr>
<tr>
<td>212</td>
<td>The Site Collection is Locked.</td>
</tr>
<tr>
<td>1816</td>
<td>The Quota for the Site Collection has been exceeded.</td>
</tr>
<tr>
<td>-2147467259</td>
<td>An error occurred while the stored procedure was running.</td>
</tr>
</tbody>
</table>
3.1.4.97 proc_UpdateWebPartTypeId

The proc_UpdateWebPartTypeId stored procedure is called to update a Web Part's Web Part type identifier property. The T-SQL syntax for the stored procedure is as follows:

PROCEDURE proc_UpdateWebPartTypeId(
    @SiteId uniqueidentifier,
    @WebPartID uniqueidentifier,
    @WebPartTypeId uniqueidentifier,
    @Assembly nvarchar(255),
    @Class nvarchar(255),
    @SolutionId uniqueidentifier,
    @RequestGuid uniqueidentifier = NULL OUTPUT
);

@SiteId: The Site Collection identifier of the site collection which contains the Web Part. MUST NOT be NULL.

@WebPartID: The Web Part identifier (as specified in [MS-WSSFO2] section 2.2.1.15) of the Web Part within the Site Collection. MUST NOT be NULL.

@WebPartTypeId: New Web Part type identifier of the Web Part. MUST NOT be NULL.

@Assembly: The fully qualified name of the assembly that implements the web part.

@Class: The name of the .NET class that implements the web part.

@SolutionId: The identifier of the sandboxed solution that implements the web part.

@RequestGuid: The optional request identifier for the current request.

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 completion.</td>
</tr>
<tr>
<td>-2147467259</td>
<td>An error occurred while the stored procedure was running.</td>
</tr>
</tbody>
</table>

Result Sets: MUST NOT return any result sets.

3.1.4.98 proc_UpdateWebPartWhileSaving

The proc_UpdateWebPartWhileSaving stored procedure is called to add a new Web Part or update an existing Web Part's properties. The T-SQL syntax for the stored procedure is as follows:

PROCEDURE proc_UpdateWebPartWhileSaving(
    @SiteId uniqueidentifier,
    @DirName nvarchar(256),
    @LeafName nvarchar(128),
    @Level tinyint,
    @WebPartID uniqueidentifier,
    @WebPartTypeID uniqueidentifier,
    @Assembly nvarchar(255),
    @Class nvarchar(255),
    @SolutionId uniqueidentifier,
@SiteId: The Site Collection identifier of the site collection which contains the requested Web Part.

@DirName: The Directory Name of the Web Part Page containing the Web Part. MUST NOT be NULL.

@LeafName: The Leaf Name of the Web Part Page containing the Web Part. MUST NOT be NULL.

@Level: The Publishing Level of the Web Part. MUST NOT be NULL.

@WebPartID: The Web Part identifier (as specified in [MS-WSSFO2], section 2.2.1.15) of the Web Part within the Site Collection. If @WebPartID matches the Web Part identifier (as specified in [MS-WSSFO2], section 2.2.1.15) of an existing Web Part on the different Web Part Page, then the protocol server MUST generate a new Web Part identifier (as specified in [MS-WSSFO2], section 2.2.1.15). MUST NOT be NULL.

@WebPartTypeID: The Web Part type identifier of the Web Part. If WebPartTypeID of the existing Web Part is changed and @IsIncluded parameter is NULL and @Level is not equal to LEVEL_CHECKOUT then personalizable properties on the Web Part MUST be deleted. MUST NOT be NULL.

@Assembly: The fully qualified name of the assembly that implements the web part.

@Class: The name of the .NET class that implements the web part.

@SolutionId: The identifier of the sandboxed solution that implements the web part.

@TheListID: The list identifier of the list associated with the web part.

@IsIncluded: The Web Part Is Closed State of the Web Part.

@FrameState: The Web Part Chrome State of the Web Part. MUST NOT be NULL.

@ZoneID: The Web Part Zone identifier of the Web Part Zone of the Web Part.

@PartOrder: The Web Part Zone Index of the Web Part.

@TheFlags: The View Flags, as specified in [MS-WSSFO2], section 2.2.12, of the Web Part.

@TheType: The web part type identifier of the web part.

@TheBaseViewID: The base view identifier of the web part.
@ContentTypeID: The Content type identifier (as specified in [MS-WSSFO2], section 2.2.1.1) of the list Items in the list to be displayed in the Web Part.

@Source: The Web Part properties of the Web Part in either WPV2:WebPart format (as specified in [MS-WPPS], section 2.2.3.2) or HTML format.

@AllUsersProperties: A binary payload containing 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.

@PerUserProperties: A binary payload containing zero or more personalizable properties on the Web Part. If this value is NULL then default values will be used for all of the personalizable properties on the Web Part.

@WebPartIdProperty: The HTML (HyperText Markup Language) ID attribute of the Web Part. May be NULL. If not NULL, it MUST be unique per Web Part Page.

@RequestGuid: The optional request identifier for the current request.

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 completion.</td>
</tr>
<tr>
<td>1</td>
<td>An error occurred executing the stored procedure.</td>
</tr>
<tr>
<td>2</td>
<td>The specified Web Part Page cannot be found.</td>
</tr>
<tr>
<td>5</td>
<td>The Web Part being updated is not on the Web Part Page.</td>
</tr>
<tr>
<td>33</td>
<td>The specified Web Part Page is not the Current Version.</td>
</tr>
<tr>
<td>212</td>
<td>The specified Site Collection is Locked.</td>
</tr>
<tr>
<td>1359</td>
<td>An internal error occurred.</td>
</tr>
<tr>
<td>1816</td>
<td>The Quota for the specified Site Collection has been exceeded.</td>
</tr>
<tr>
<td>-2147467259</td>
<td>An error occurred while the stored procedure was running.</td>
</tr>
</tbody>
</table>

Result Sets: MUST NOT return any result sets.

3.1.4.99 proc_UpdateWorkflowAssociation

The proc_UpdateWorkflowAssociation stored procedure is called to update a workflow association. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_UpdateWorkflowAssociation(
    @Id uniqueidentifier,
    @SiteId uniqueidentifier,
    @Name nvarchar(255),
    @Description nvarchar(1023),
    @StatusFieldName nvarchar(64),
    @TaskListId varbinary(16),
    @HistoryListId varbinary(16),
    @TaskListTitle nvarchar(255),
    @HistoryListTitle nvarchar(255),
)
```
@Configuration            int,
@AutoCleanupDays          int,
@PermissionsManual        bigint,
@InstantiationParams      nvarchar(max),
@Version                  int,
@RequestGuid              uniqueidentifier = NULL OUTPUT
);

@Id: The Workflow association identifier of the Workflow association being updated. The server MUST update the modification date and time of the Workflow association to the date and time in UTC when the stored procedure was called.

@SiteId: The Site Collection identifier of the Site Collection which contains the Workflow association.

@Name: The name of the Workflow association. If this value is NULL, the server MUST NOT update the name field of the Workflow association.

@Description: The description of the Workflow association. If this value is NULL, the server MUST NOT update the description field of the Workflow association.

@StatusFieldName: The name of the Workflow Status field of the Workflow association. If this value is NULL, the server MUST NOT update the Workflow Status field of the Workflow association.

@TaskListId: The list identifier of the Workflow Task list of the Workflow association. If this value is NULL, the server MUST NOT update the Workflow Task list identifier field of the Workflow association.

@HistoryListId: The list identifier of the Workflow History list of the Workflow association. If this value is NULL, the server MUST NOT update the Workflow History list identifier field of the Workflow association.

@TaskListTitle: The title of the Workflow Task list of the Workflow association. If this value is NULL, the server MUST NOT update the Workflow Task list title field of the Workflow association.

@HistoryListTitle: The title of the Workflow History list of the Workflow association. If this value is NULL, the server MUST NOT update the Workflow History list title field of the Workflow association.

@Configuration: The Workflow association Configuration of the Workflow association. This value MUST NOT be NULL.

@AutoCleanupDays: The number of days before Workflows based on the Workflow association are cleaned up. This value MUST contain a positive integer.

@PermissionsManual: The rights mask (as specified in [MS-WSSFO2], section 2.2.2.14) required to manually start any Workflows created from the Workflow association. This value MUST NOT be NULL.

@InstantiationParams: The Workflow association Data of the Workflow association. If this value is NULL, the server MUST NOT update the workflow association data of the Workflow association.

@Version: MUST contain either the value 0 or the current version of the workflow association specified by @Id. The server MUST increment the version of the Workflow association by 1.

@RequestGuid: The optional request identifier for the current request.

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 completion.</td>
</tr>
<tr>
<td>5</td>
<td>An error occurred.</td>
</tr>
</tbody>
</table>

Result Sets: MUST NOT return any result sets.

### 3.1.4.100  proc_UpdateWorkItem

The proc_UpdateWorkItem stored procedure is called to modify the properties of a Work Item. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_UpdateWorkItem(
    @WorkItemId uniqueidentifier,
    @SiteId uniqueidentifier,
    @DeliveryDate datetime,
    @BinaryPayload varbinary(max),
    @TextPayload nvarchar(max),
    @ProcessingId uniqueidentifier,
    @ForceUpdate bit= 0,
    @RequestGuid uniqueidentifier = NULL OUTPUT
);
```

**@WorkItemId:** The Work Item identifier. The server MUST only update a Work Item if it has the given Work Item identifier. MUST NOT be NULL.

**@SiteId:** The Site Collection identifier of the Site Collection. The server MUST only update a Work Item if it is associated with this Site Collection. MUST NOT be NULL.

**@DeliveryDate:** The Work Item Delivery Date. If the parameter is NULL, then the server MUST NOT change the Work Item Delivery Date associated with the Work Item. If the parameter is not NULL, then the server MUST update the Delivery Date of the Work Item to this value. In this case, if the parameter value differs from the previous Delivery Date, then the server MUST:

- Set the Work Item Processing identifier associated with the Work Item to NULL,
- Mark the Work Item as not In Progress Work Item, and
- Mark the Work Item as not Throttled Fetch.

**@BinaryPayload:** The work item binary payload.

**@TextPayload:** The work item text payload.

**@ProcessingId:** The Work Item Processing identifier of the Work Item Process. If the value of @ForceUpdate is 0, then the server MUST only modify Work Items associated with the Work Item Process indicated by this parameter.

**@ForceUpdate:** Specifies whether or not the stored procedure MUST update Work Items that do not have the same Work Item Process as the one specified by @ProcessingId. If the value of the parameter is 0, then the server MUST only update the Work Item if the @ProcessingId parameter matches the Work Item Processing identifier associated with the Work Item. If the value of the parameter is 1, then the server MUST update the Work Item regardless of the value of the @ProcessingId parameter.
@RequestGuid: The optional request identifier for the current request.

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 completion.</td>
</tr>
<tr>
<td>5</td>
<td>Error: Access denied.</td>
</tr>
</tbody>
</table>

Result Sets: MUST NOT return any result sets.

3.1.4.101 proc_WorkflowHasVisibleParentItem

The proc_WorkflowHasVisibleParentItem stored procedure is called to determine if the list Item that the Workflow was created for has been deleted. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_WorkflowHasVisibleParentItem(
    @SiteId                  uniqueidentifier,
    @WorkflowInstanceId      uniqueidentifier,
    @RequestGuid             uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The Site Collection identifier of the Site Collection which contains the Workflow.

@WorkflowInstanceId: The Workflow identifier of the Workflow.

@RequestGuid: The optional request identifier for the current request.

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 list Item that the Workflow was created for has been deleted.</td>
</tr>
<tr>
<td>1</td>
<td>The list Item that the Workflow was created for has not been deleted.</td>
</tr>
</tbody>
</table>

Result Sets: MUST NOT return any result sets.

3.1.5 Timer Events

If the timeout event is triggered, the stored procedure is terminated and the call fails.

3.1.6 Other Local Events

None.

3.2 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

Refer to section 3.1.1.

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 (as specified in [MS-WSSFO2], section 2.2.1.9) and the User identifier (as specified in [MS-WSSFO2], section 2.2.1.13) 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 Return 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 Database or Configuration databases, unless explicitly described in this section.

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 affect the operation of this protocol.
4  Protocol Examples

This section provides specific example scenarios for manipulating Event receivers, Web Parts, Workflows, and Work Items. These examples describe in detail the process of communication between the front-end Web server and the back end database server.

4.1  Event Receiver

4.1.1  Create an Event Receiver

This example describes the request made and the response returned when a user registers a new event receiver to handle an event for a list in a site.

The user initiates this scenario by registering the new event receiver for the list as specified in the following figure.

![Diagram of Create an event receiver](image)

**Figure 2: Create an event receiver**

1. The user creates a list object that represents the event host list and adds the new event receiver for the desired event to the event host list.

2. The front-end Web server calls the `proc_InsertEventReceiver` stored procedure to save the event receiver registration to back end database server.

3. The `proc_InsertEventReceiver` stored procedure returns a return code.

4. The control returns to the user.

4.1.2  Read Event Receivers

Reading event receivers is part of reading the metadata of the event host. Please refer to [MS-WSSFO2], section 4.4, for an example.

4.1.3  Update an Event Receiver

This scenario is initiated by a user who wants to update an event receiver for a list.

![Diagram of Update an event receiver](image)
Figure 3: Update an event receiver

1. The user creates a list object that represents the event host list, gets the event receiver to update via the event host list, modifies the properties of the event receiver as desired, then updates the event receiver.

2. The front-end Web server calls the `proc_UpdateEventReceiver` stored procedure to save the event receiver's new properties to the back end database server.

3. The `proc_UpdateEventReceiver` stored procedure returns a return code.

4. The control returns to the user.

4.1.4 Delete an Event Receiver

This scenario is initiated by a user who wants to delete an event receiver from a list.

Figure 4: Delete an event receiver

1. The user creates a list object that represents the event host list, gets the event receiver to delete via the event host list, then deletes the event receiver.

2. The front-end Web server calls the `proc_DeleteEventReceiver` stored procedure to delete the event receiver in the back end database server.

3. The `proc_DeleteEventReceiver` stored procedure returns a return code.

4. The control returns to the user.

4.2 Web Part

4.2.1 Add a List View Web Part

This scenario is initiated when a list View Web Part is added to a Web Part Page.
Figure 5: Add a List View Web Part

This example assumes that the list View Web Part to be added is instantiated and refers a valid list and a valid View associated with the list.

The following actions happen:

The front-end Web server retrieves security permissions information about the requested Site. It does this by calling the \[MS-WSSFO2\].proc_SecGetSecurityInfo stored procedure (as specified in \[MS-WSSFO2\] section 3.1.5.84).
The back-end database server returns the Security Information Result Set, which consists of information about security permissions about the requested Site.

The front-end Web server requests information about the Web Part Page to which the list View Web Part needs to be added by calling the [MS-WSSFO2].proc_FetchDocForRead stored procedure (as specified in [MS-WSSFO2] section 3.1.5.20).

The Back-End Database Server returns a set of Result Sets as defined in [MS-WSSFO2] section 3.1.5.17.1 through [MS-WSSFO2] section 3.1.5.17.3, and the Publishing Level of the Document as an output parameter.

The front-end Web server then fetches properties of the list that the list View Web Part refers to by calling the [MS-WSSFO2].proc_GetListMetaDataAndEventReceivers stored procedure (as specified in [MS-WSSFO2] section 3.1.5.34).

The Back-End Database Server returns two Result Sets which include the metadata and Event Receivers for the specified list.

The front-end Web server then fetches Views associated with the list by calling the proc_GetListWebParts stored procedure.

The Back-End Database Server returns one Result Set which include the list views associated with the list.

The front-end Web server sends a request to create a new list View Web Part and a new associated View for the list and places the list View Web Part in the specified Web Part Page by calling the proc_CreateListViewPart stored procedure.

The Back-End Database Server returns an output code and the Publishing Level as the output parameter.

The front-end Web server then re-fetches the Views corresponding to the list by calling the proc_GetListWebParts stored procedure.

The Back-End Database Server returns one Result Set which include the list Views corresponding to the list.

The front-end Web server then sends a request to copy properties of the View specified in the list View Web Part instance to the newly created View by calling the proc_ApplyViewToListWebPart stored procedure.

The Back-End Database Server returns a Return Code status and the View Flags (as specified in [MS-WSSFO2], section 2.2.2.12) of the new View for the Web Part as an output parameter.

4.2.2 Add a non-List View Web Part

This scenario is initiated when a Web Part which is NOT a list View Web Part is added to a Web Part Page.
Figure 6: Add a non-List View Web Part

This example assumes the Web Part to be added is instantiated. The following actions happen:

1. The front-end Web server retrieves security permissions information about the requested Site. It does this by calling the [MS-WSSFO2].proc_SecGetSecurityInfo stored procedure (as specified in [MS-WSSFO2] section 3.1.5.84).

2. The Back-End Database Server returns the Security Information Result Set, which consists of information about security permissions about the requested Site.

3. The front-end Web server requests information about the Web Part Page to which the Web Part needs to be added by calling the [MS-WSSFO2].proc_FetchDocForRead stored procedure (as specified in [MS-WSSFO2] section 3.1.5.20).

4. The Back-End Database Server returns a set of Result Sets as defined in [MS-WSSFO2] section 3.1.5.17.1 through [MS-WSSFO2] section 3.1.5.17.3, and the Publishing Level of the Document as an output parameter.

5. The front-end Web server sends a request to create a new Web Part and place it in the specified Web Part Zone on the specified Web Part Page by calling the proc_AddWebPart stored procedure.

6. The Back-End Database Server returns an output code and the Publishing Level as the output parameter.

4.2.3 Get All Web Parts on a Web Part Page

This scenario is initiated when a request is made to fetch all the Web Parts on a Web Part Page.
Figure 7: Retrieve all Web Parts on a Web Part Page

The following actions happen:


4.2.4 Delete a Web Part

This scenario is initiated when a Web Part is deleted from a Web Part Page.

This example assumes:

- The Web Part to be deleted is on the specified Web Part Page.
- The Web Part is not personalized.
- The Web Part Page is not contained in a Document Library, or the Document Library containing the Web Part Page has **Required Checkout** set to 0.

The following actions happen:

1. The front-end Web server builds a dynamic T-SQL syntax query which requests the particular Web Part to be deleted by calling `proc_DeleteWebPart` stored procedure. It also queries the return code and the output Publishing Level of the Document from the stored procedure.

4.3 Workflow

4.3.1 Create a Workflow for a List Item

This scenario is initiated when a Workflow is added to a list item.

![Create a Workflow for a List Item Diagram](image)

**Figure 9: Create a Workflow for a List Item**

This example assumes the Workflow to be added refers to a valid list Item and Workflow association associated with the parent list. The following actions happen:

1. The front-end Web server sends a request to create a new Workflow on the specified list Item by calling the `proc_AddWorkflow` stored procedure.
2. The Back-End Database Server returns a return code specifying the outcome.

4.3.2 Delete a Workflow from a List Item

This scenario is initiated when a Workflow is removed from a list item.

![Delete a Workflow from a List Item Diagram](image)

**Figure 10: Delete a Workflow from a List Item**

This example assumes the Workflow to be removed is instantiated and refers to a valid list Item. The following actions happen:

1. The front-end Web server sends a request to delete an existing Workflow on the specified list Item by calling the `proc_DropWorkflow` stored procedure.
2. The Back-End Database Server returns a return code specifying the outcome.

4.4 Work Item

4.4.1 Create a Work Item for Bulk Editing Workflow Tasks

This scenario is initiated when a user clicks on a button in the client UI to bulk edit workflow tasks with a certain set of values.
Figure 11: Create a Work Item for Bulk Editing Workflow Tasks

The following actions happen:

1. The front-end Web server requests to create a new Work Item by calling the `proc_AddWorkItem` stored procedure specifying a work item type identifier representing bulk workflow tasks and a Work Item Delivery Date of the current time to indicate that the work item executes as soon as possible.

2. The Back-End Database Server creates a new Work Item in the Content Database and returns a single Return Code status to indicate whether the Work Item was successfully created.

4.4.2 Retrieve a Set of Runnable Bulk Workflow Task Work Items

This scenario is initiated when a Timer Job runs that executes Work Items of Work Item type Bulk Workflow Task.

Figure 12: Retrieve a Set of Runnable Bulk Workflow Task Work Items

This example assumes that the Content Database already contains work items whose Delivery dates have passed and are of Work item type bulk workflow task. The following actions happen:

1. The front-end Web server requests the set of Work Items which have delivery dates at or before the current time and are of work item type bulk workflow task by calling the `proc_GetRunnableWorkItems` stored procedure.

2. The Back-End Database Server returns a set of Work Items and marks them as In Progress Work Items. The Timer Job can then iterate through and run all Work Items in the set.

4.4.3 Delete a Work Item

This scenario is initiated when the Timer Job has completed execution of a Work Item that have delivery dates that have passed and is about to mark them as completed.
**Figure 13: Delete a Work Item**

The following actions happen:

1. The front-end Web server requests to mark Work Items as completed by calling the `proc_DropWorkItem` stored procedure.

2. The Back-End Database Server deletes the work item and returns a single Return Code status to indicate execution completion.
5 Security

5.1 Security Considerations for Implementers

Interactions with SQL are susceptible to tampering and other forms of security risks. Implementers are advised to sanitize input parameters for stored procedures prior to calling the stored procedure. Refer to [MS-SPPTC] for security considerations relating to sandboxed solutions.

5.2 Index of Security Parameters

None.
6 Appendix A: Product Behavior

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include released service packs:

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

Exceptions, if any, are noted below. If a service pack or Quick Fix Engineering (QFE) number appears with the product version, behavior changed in that service pack or QFE. The new behavior also applies to subsequent service packs of the product unless otherwise specified. If a product edition appears with the product version, behavior is different in that product edition.

Unless otherwise specified, any statement of optional behavior in this specification that is prescribed using the terms SHOULD or SHOULD NOT implies product behavior in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term MAY implies that the product does not follow the prescription.

<1> Section 2.2.6.6: Section 2.2.5.4: SharePoint Products and Technologies MAY use 1 as an arbitrary placeholder when there is no list item associated with the work item.

<2> Section 3.1.4.7: Section 3.1.4.6: SharePoint Products and Technologies MAY use 1 as an arbitrary placeholder when there is no list item associated with the work item.
7 Change Tracking

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

## A

Abstract data model
- client 139
- server 28

Add a list view Web Part example 141
Add a non-list view Web Part example 143

Applicability 13

Attribute groups - overview 27
Attributes - overview 27

## B

Binary structures - overview 19

Bit fields
- event receiver source type bit field 16
- workflow association configuration 17
- workflow internal state 18
- Workflow Status1 18

## C

Capability negotiation 13
Change tracking 151

Client
- abstract data model 139
- Content Database Programmability Extensions Communications interface 138
- initialization 139
- local events 139
- message processing 139
- overview 138
- sequencing rules 139
- timer events 139
- timers 139

Common data types
- overview 16

Complex types - overview 26
- Content Database Programmability Extensions Communications interface 138
- Context collection identifier simple type 16
- Context identifier simple type 16
- Context object identifier simple type 16
- Create a work item for bulk editing workflow tasks example 146
- Create a workflow for a list item example 146
- Create an event receiver example 140

## D

Data model - abstract
- client 139
- server 28

Data types
- common 16
- context collection identifier simple type 16
- context identifier simple type 16
- context object identifier simple type 16
- context type identifier simple type 16
- event receiver source identifier simple type 16
- list item version simple type 16
- workflow template identifier simple type 16

Data types - simple
- context identifier 16
- context object identifier 16
- context type identifier 16
- event receiver source identifier 16
- context collection identifier 16
- workflow template identifier 16

Data types = simple
- list item version 16

Delete a Web Part example 145
Delete a work item example 147
Delete a workflow from a list item example 146
Delete an event receiver example 141

## E

Elements
- Workflow Modifications 26
- Elements - overview 26

Enumerations
- sandboxed solution status 19

Event operations overview 11

Event receiver source identifier simple type 16

Event receiver source type bit field 16

Events
- local - client 139
- local - server 138
- timer - client 139
- timer - server 138

Examples
- add a list view Web Part 141
- add a non-list view Web Part 143
- create a work item for bulk editing workflow tasks 146
- create a workflow for a list item 146
- create an event receiver 140
- delete a Web Part 145
- delete a workflow 147
- delete a workflow from a list item 146
- delete an event receiver 141
- get all Web Parts on a Web Part page 144
- overview 140
- read event receivers 140
- retrieve a set of runnable bulk workflow task work items 147
- update an event receiver example 140

## F

Fields - vendor-extensible 13

## G

Get all Web Parts on a Web Part page example 144

Glossary 8
<table>
<thead>
<tr>
<th>Method Name</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>proc_ProcessSolutionResourceUsageWindowedData</td>
<td>106</td>
</tr>
<tr>
<td>proc_ProvisionWebPart</td>
<td>107</td>
</tr>
<tr>
<td>proc_RemoveSolution</td>
<td>108</td>
</tr>
<tr>
<td>proc_ResetSiteResourceUsageWarnings</td>
<td>108</td>
</tr>
<tr>
<td>proc_RestoreWebPartForDoc</td>
<td>108</td>
</tr>
<tr>
<td>proc_RevertInProgressWorkItem</td>
<td>109</td>
</tr>
<tr>
<td>proc_RevertInProgressWorkItems</td>
<td>110</td>
</tr>
<tr>
<td>proc_SetEventReceiverToSynchronous</td>
<td>111</td>
</tr>
<tr>
<td>proc_TruncateResourceUsageDaily</td>
<td>111</td>
</tr>
<tr>
<td>proc_TruncateResourceUsageLog</td>
<td>112</td>
</tr>
<tr>
<td>proc_TruncateResourceUsageWindowed</td>
<td>112</td>
</tr>
<tr>
<td>proc_UpdateDataViewWhileSaving</td>
<td>112</td>
</tr>
<tr>
<td>proc_UpdateDocEventReceiver</td>
<td>113</td>
</tr>
<tr>
<td>proc_UpdateEventReceiver</td>
<td>115</td>
</tr>
<tr>
<td>proc_UpdateListItemWorkflowInstanceStateData</td>
<td>118</td>
</tr>
<tr>
<td>proc_UpdateListItemWorkflowLock</td>
<td>121</td>
</tr>
<tr>
<td>proc_UpdateListItemWorkflowSource</td>
<td>122</td>
</tr>
<tr>
<td>proc_UpdateListViewDataViewForSite</td>
<td>122</td>
</tr>
<tr>
<td>proc_UpdateListViewDataViewForWeb</td>
<td>123</td>
</tr>
<tr>
<td>proc_UpdateSiteResourceUsage</td>
<td>123</td>
</tr>
<tr>
<td>proc_UpdateSolutionResourceUsage</td>
<td>124</td>
</tr>
<tr>
<td>proc_UpdateViewWhileSaving</td>
<td>125</td>
</tr>
<tr>
<td>proc_UpdateWebPart</td>
<td>127</td>
</tr>
<tr>
<td>proc_UpdateWebPartCache</td>
<td>129</td>
</tr>
<tr>
<td>proc_UpdateWebPartIsIncluded</td>
<td>130</td>
</tr>
<tr>
<td>proc_UpdateWebPartProps</td>
<td>131</td>
</tr>
<tr>
<td>proc_UpdateWebPartTypeld</td>
<td>133</td>
</tr>
<tr>
<td>proc_UpdateWebPartWhileSaving</td>
<td>133</td>
</tr>
<tr>
<td>proc_UpdateWorkflowAssociation</td>
<td>135</td>
</tr>
<tr>
<td>proc_UpdateWorkflowItem</td>
<td>137</td>
</tr>
<tr>
<td>proc_WorkflowHasVisibleParentItem</td>
<td>138</td>
</tr>
<tr>
<td>proc_CancelWorkflow method</td>
<td>46</td>
</tr>
<tr>
<td>proc_CommitUpdatedZoneIds method</td>
<td>48</td>
</tr>
<tr>
<td>proc_CompleteInProgressWorkItems method</td>
<td>48</td>
</tr>
<tr>
<td>proc_CopyDefaultViewWebParts method</td>
<td>49</td>
</tr>
<tr>
<td>proc_CountWorkflowAssociations method</td>
<td>50</td>
</tr>
<tr>
<td>proc_CountWorkflows method</td>
<td>50</td>
</tr>
<tr>
<td>proc_CountWorkflowsBatch method</td>
<td>51</td>
</tr>
<tr>
<td>proc_CreateListViewPart method</td>
<td>52</td>
</tr>
<tr>
<td>proc_DeleteDocEventReceiver</td>
<td>54</td>
</tr>
<tr>
<td>proc_DeleteEventReceiver</td>
<td>56</td>
</tr>
<tr>
<td>proc_DeleteEventReceiversBySourceId method</td>
<td>57</td>
</tr>
<tr>
<td>proc_DeleteInProgressWorkItems method</td>
<td>58</td>
</tr>
<tr>
<td>proc_DeleteWebPart method</td>
<td>60</td>
</tr>
<tr>
<td>proc_DeleteWebPartPersonalization method</td>
<td>61</td>
</tr>
<tr>
<td>proc_DeleteWebPartWhileSaving</td>
<td>62</td>
</tr>
<tr>
<td>proc_DeleteZoneWebPartsWhileSaving method</td>
<td>63</td>
</tr>
<tr>
<td>proc_DisableAssociationsForTemplate method</td>
<td>63</td>
</tr>
<tr>
<td>proc_DeleteWorkflow method</td>
<td>64</td>
</tr>
<tr>
<td>proc_DeleteWorkflowAssociation method</td>
<td>64</td>
</tr>
<tr>
<td>proc_DeleteWorkflowItem method</td>
<td>65</td>
</tr>
<tr>
<td>proc_EnableDeclarativeWorkflowAssociations method</td>
<td>65</td>
</tr>
<tr>
<td>proc_EmumerateWebPartsForList method</td>
<td>66</td>
</tr>
<tr>
<td>proc_EnumerateWebPartsForWeb method</td>
<td>66</td>
</tr>
<tr>
<td>proc_EnumResourceWarningSites method</td>
<td>68</td>
</tr>
<tr>
<td>proc_FailOverInProgressWorkItems method</td>
<td>69</td>
</tr>
<tr>
<td>proc_GetAllResourceUsageForSiteToday method</td>
<td>70</td>
</tr>
<tr>
<td>proc_GetAllWebPartsOnPage method</td>
<td>70</td>
</tr>
<tr>
<td>proc_GetAverageDailyResourceUsageForSite method</td>
<td>72</td>
</tr>
<tr>
<td>proc_GetContextCollectionEventReceivers method</td>
<td>73</td>
</tr>
<tr>
<td>proc_GetContextObjectEventReceivers method</td>
<td>73</td>
</tr>
<tr>
<td>proc_GetListWebParts method</td>
<td>74</td>
</tr>
<tr>
<td>proc_GetListWorkflowItemsWithInstanceDataAndLock</td>
<td>75</td>
</tr>
<tr>
<td>proc_GetListWebParts method</td>
<td>77</td>
</tr>
<tr>
<td>proc_GetNextWebPartOrder method</td>
<td>79</td>
</tr>
<tr>
<td>proc_GetRecycleBinItemEventReceivers method</td>
<td>80</td>
</tr>
<tr>
<td>proc_GetRunnableWorkflowItems method</td>
<td>81</td>
</tr>
<tr>
<td>proc_GetRunningWorkflowEventReceivers method</td>
<td>83</td>
</tr>
<tr>
<td>proc_GetSiteResourceUsage method</td>
<td>83</td>
</tr>
<tr>
<td>proc_GetSolutionInfo method</td>
<td>84</td>
</tr>
<tr>
<td>proc_GetSolutionInfo method</td>
<td>84</td>
</tr>
<tr>
<td>proc_GetSolutionResourceQuota method</td>
<td>85</td>
</tr>
<tr>
<td>proc_GetSolutionResourceUsage method</td>
<td>86</td>
</tr>
<tr>
<td>proc_GetSolutionResourceUsageDailyOrdnal method</td>
<td>86</td>
</tr>
<tr>
<td>proc_GetSolutionsData method</td>
<td>87</td>
</tr>
<tr>
<td>proc_GetWFTemplatesLastModifiedForWeb method</td>
<td>88</td>
</tr>
<tr>
<td>proc_GetWorkflowAssociations method</td>
<td>88</td>
</tr>
<tr>
<td>proc_GetWorkflowDataForItem method</td>
<td>89</td>
</tr>
<tr>
<td>proc_GetWorkItems method</td>
<td>90</td>
</tr>
<tr>
<td>proc_InsertContextEventReceiver method</td>
<td>91</td>
</tr>
<tr>
<td>proc_InsertDocEventReceiver method</td>
<td>94</td>
</tr>
<tr>
<td>proc_InsertEventReceiver method</td>
<td>95</td>
</tr>
<tr>
<td>proc_LogSolutionResourceUsage20 method</td>
<td>97</td>
</tr>
</tbody>
</table>
Retrieve a set of runnable bulk workflow task work items example 147

S
Sandboxed solution status enumeration 19
Security
  implementer considerations 149
  parameter index 149
Sequencing rules
  client 139
  server 31
Server
  abstract data model 28
  initialization 31
  local events 138
  message processing 31
  proc_AddNonListViewFormPersonalization method 31
  proc_AddNonListViewFormWebPartForUrl method 33
  proc_AddSolution method 35
  proc_AddWebPart method 36
  proc_AddWorkflow method 38
  proc_AddWorkflowAssociation method 40
  proc_AddWorkItem method 41
  proc_ApplyViewToListWebPart method 43
  proc_AutoCleanupWorkflows method 45
  proc_AutoDropWorkflows method 47
  proc_CancelDeclarativeWorkflows method 47
  proc_CancelWorkflow method 47
  proc_CommitUpdatedZoneIds method 48
  proc_CompleteInProgressWorkItems method 48
  proc_CopyDefaultViewWebParts method 49
  proc_CountWorkflowAssociations method 50
  proc_CountWorkflows method 50
  proc_CountWorkflowsBatch method 51
  proc_CreateListViewPart method 52
  proc_DeleteDocEventReceiver method 54
  proc_DeleteEventReceiver method 56
  proc_DeleteEventReceiversBySourceId method 57
  proc_DeleteInProgressWorkItems method 58
  proc_DeleteSmartPagePersonalization method 59
  proc_DeleteWebPart method 60
  proc_DeleteWebPartPersonalization method 61
  proc_DeleteWebPartWhileSaving method 62
  proc_DeleteZoneWebPartsWhileSaving method 63
  proc_DisableAssociationsForTemplate method 63
  proc_DropWorkflow method 64
  proc_DropWorkflowAssociation method 64
  proc_DropWorkItem method 65
  proc_EnableDeclarativeWorkflowAssociations method 65
  proc_EnumerateWebPartsForList method 66
  proc_EnumerateWebPartsForWeb method 66
  proc_EnumResourceWarningSites method 68
  proc_FailOverInProgressWorkItems method 69
  proc_GetAllResourceUsageForSiteToday method 70
  proc_GetAllWebPartsOnPage method 70

R
Read event receivers example 140
References 10
  informative 11
  normative 11
Relationship to other protocols 12
Result sets - messages
  List Item Workflows 19
  Solution Resource Usage 21
  Solution Resource Usage Processing 21
  Web Parts 22
  Work Items 25
  Workflow Associations 23