[MS-UPSCHNG2]:
User Profile Change Log Stored Procedure Version 2 Protocol Specification

---

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation for protocols, file formats, languages, standards as well as overviews of the interaction among each of these technologies.
- **Copyrights.** This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you may make copies of it in order to develop implementations of the technologies described in the Open Specifications and may distribute portions of it in your implementations using these technologies or your documentation as necessary to properly document the implementation. You may also distribute in your implementation, with or without modification, any schema, IDL's, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications.
- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.
- **Patents.** Microsoft has patents that may cover your implementations of the technologies described in the Open Specifications. Neither this notice nor Microsoft's delivery of the documentation grants any licenses under those or any other Microsoft patents. However, a given Open Specification may be covered by Microsoft's Open Specification Promise (available here: http://www.microsoft.com/interop/osp) or the Community Promise (available here: http://www.microsoft.com/interop/cp/default.mspx). If you would prefer a written license, or if the technologies described in the Open Specifications are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **Trademarks.** The names of companies and products contained in this documentation may be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights.
- **Fictitious Names.** The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

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

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

Revision Summary

Date	Revision History	Revision Class	Comments
07/13/2009	0.1	Major	Initial Availability
08/28/2009	0.2	Editorial	Revised and edited the technical content
11/06/2009	0.3	Editorial	Revised and edited the technical content
02/19/2010	1.0	Major	Updated and revised the technical content
03/31/2010	1.01	Editorial	Revised and edited the technical content
04/30/2010	1.02	Editorial	Revised and edited the technical content
06/07/2010	1.03	Editorial	Revised and edited the technical content
06/29/2010	1.04	Editorial	Changed language and formatting in the technical content.
07/23/2010	1.05	Minor	Clarified the meaning of the technical content.
09/27/2010	1.05	No change	No changes to the meaning, language, or formatting of the technical content.
11/15/2010	1.05	No change	No changes to the meaning, language, or formatting of the technical content.
12/17/2010	1.06	Editorial	Changed language and formatting in the technical content.

Table of Contents

1 Introduction 6
1.1 Glossary 6
1.2 References 7
1.2.1 Normative References 7
1.2.2 Informative References 7
1.3 Protocol Overview (Synopsis) 8
1.4 Relationship to Other Protocols 9
1.5 Prerequisites/Preconditions 10
1.6 Applicability Statement 10
1.7 Versioning and Capability Negotiation 10
1.8 Vendor-Extensible Fields 10
1.9 Standards Assignments 10

2 Messages 11
2.1 Transport 11
2.2 Common Data Types 11
2.2.1 Simple Data Types and Enumerations 11
2.2.1.1 Privacy Policy Type 11
2.2.1.2 Change Type 11
2.2.1.3 Object Type 11
2.2.1.4 Value 12
2.2.1.5 Privacy Type 13
2.2.2 Bit Fields and Flag Structures 13
2.2.3 Binary Structures 13
2.2.4 Result Sets 13
2.2.4.1 GetUserColleagueEvents 13
2.2.4.2 GetUserEvents 15
2.2.4.3 GetUserEventsNoRecordId 15
2.2.4.4 profile_EnumProfileChanges.profile_EnumProfileChanges.Default.ResultSet0 16
2.2.4.5 GetCurrentChangeTokenUser 17
2.2.4.6 GetInterestsExpertsEvents 17
2.2.4.7 GetUserEventsRecordId 18
2.2.4.8 profile_EnumProfileChanges.profile_EnumProfileChanges.Default1.ResultSet0 19
2.2.4.9 GetCurrentChangeTokenOrganization 19
2.2.4.10 GetActivityApplications 20
2.2.4.11 GetActivityEvents 20
2.2.4.12 GetActivityPreferences 21
2.2.4.13 GetActivityTemplates 21
2.2.4.14 GetActivityTypes 22
2.2.4.15 GetLatestLastActivityEventUpdateTime 23
2.2.4.16 GetUserInfo 23
2.2.4.17 GetUsersColleaguesAndRights 24
2.2.4.18 GetUserEventsIDOldest 24
2.2.4.19 GetUserEventsIDNewest 24
2.2.5 Tables and Views 25
2.2.6 XML Structures 25
2.2.6.1 Namespaces 25
2.2.6.2 Simple Types 25
2.2.6.3 Complex Types 25
2.2.6.3.1 Link 25

2.2.6.3.2 List 26
2.2.6.3.3 Entity 26
2.2.6.4 Elements 27
2.2.6.4.1 ActivityTemplateVariable 27
2.2.6.5 Attributes 28
2.2.6.6 Groups 28
2.2.6.7 Attribute Groups 28

3 Protocol Details 29
3.1 Server Details 29
3.1.1 Abstract Data Model 29
3.1.2 Timers 31
3.1.3 Initialization 31
3.1.4 Higher-Layer Triggered Events 31
3.1.5 Message Processing Events and Sequencing Rules 31
3.1.5.1 profile_GetUserEvents 31
3.1.5.2 profile_DeleteUserEvents 33
3.1.5.3 profile_GetUserColleagueEvents 33
3.1.5.4 profile_GenerateAnniversaryEvents 34
3.1.5.5 profile_GetCurrentChangeToken 35
3.1.5.6 profile_AddUserProfileEvent 35
3.1.5.7 profile_EnumProfileChanges 36
3.1.5.8 profile_GetInterestsExpertsEvents 37
3.1.5.9 profile_GetUserColleagueEventsByDate 38
3.1.5.10 activityFeed_DeleteActivityEventsConsolidated 38
3.1.5.11 activityFeed_DeleteActivityEventsPublished 39
3.1.5.12 activityFeed_DeleteActivityTemplates 40
3.1.5.13 activityFeed_DeleteActivityTypes 40
3.1.5.14 activityFeed_DeleteIfExistsActivityPreference 40
3.1.5.15 activityFeed_GetActivityApplications 41
3.1.5.16 activityFeed_GetActivityEventsConsolidated 42
3.1.5.17 activityFeed_GetActivityEventsPublished 42
3.1.5.18 activityFeed_GetActivityPreferences 44
3.1.5.19 activityFeed_GetActivityTemplates 44
3.1.5.20 activityFeed_GetActivityTypes 45
3.1.5.21 activityFeed_GetLatestLastUpdateTimeConsolidated 46
3.1.5.22 activityFeed_GetLatestLastUpdateTimePublished 46
3.1.5.23 activityFeed_InsertActivityApplications 47
3.1.5.24 activityFeed_InsertActivityEventsConsolidated 47
3.1.5.25 activityFeed_InsertActivityEventsPublished 48
3.1.5.26 activityFeed_InsertActivityTemplates 49
3.1.5.27 activityFeed_InsertActivityTypes 50
3.1.5.28 activityFeed_InsertIfNotExistsActivityPreference 52
3.1.5.29 activityFeed_UpdateActivityApplications 52
3.1.5.30 activityFeed_UpdateActivityEventsConsolidated 53
3.1.5.31 activityFeed_UpdateActivityEventsPublished 54
3.1.5.32 activityFeed_UpdateActivityTemplates 54
3.1.5.33 activityFeed_UpdateActivityTypes 55
3.1.5.34 profile_GetUsersColleaguesAndRights 57
3.1.5.35 profile_GetViewerPublisherInfo 57
3.1.5.36 profile_GetUserEventsIDRange 58
3.1.6 Timer Events 58
3.1.7 Other Local Events 58

3.2 Client Details 58
3.2.1 Abstract Data Model 58
3.2.2 Timers 58
3.2.3 Initialization 59
3.2.4 Higher-Layer Triggered Events 59
3.2.5 Message Processing Events and Sequencing Rules 59
3.2.6 Timer Events 59
3.2.7 Other Local Events 59

4 Protocol Examples 60
4.1 Retrieve All Changes for the Colleagues of a User 61
4.2 Retrieve All Changes for a Specified User 62
4.3 Synchronization 63
4.4 Recording a User Profile Change Event in the Activity Feed 64
4.5 Retrieving Information from a User's Published Feed 67
4.6 Retrieving Information from a User's Consolidated Feed 70
4.7 Delete old activities from the consolidated and published feeds 71

5 Security 73
5.1 Security Considerations for Implementers 73
5.2 Index of Security Parameters 73

6 Appendix A: Product Behavior 74

7 Change Tracking 75

8 Index 76

1 Introduction

This document specifies the User Profile Change Log Stored Procedure Protocol. This protocol allows multiple ways for the protocol client to interact with the events that are generated by the protocol server for property changes in user profiles.

1.1 Glossary

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

checksum
Coordinated Universal Time (UTC)
distinguished name (DN)
domain
globally unique identifier (GUID)
Security Support Provider Interface (SSPI)

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

change log
change token
colleague
datetime
distribution list
event
front-end Web server
HTML (HyperText Markup Language)
membership
organization
partition identifier
profile subtype
request identifier
result set
return code
Session Initiation Protocol (SIP)
site
SQL (Structured Query Language)
stored procedure
T-SQL (Transact-Structured Query Language)
URL (Uniform Resource Locator)
user profile change event
XML namespace
XML namespace prefix

The following terms are specific to this document:

activity application: An application that aggregates data, such as social data, from one or more sources and organizes that data into activity feeds.

activity event: A specific record in an activity feed. It indicates what change was made, when the change was made, and who made the change.

activity template: An XML structure that is associated with a specific activity type. It defines the structure and format for representing an activity event in each of the languages that are supported by the protocol server.

activity type: A type of activity event, such as changes to a colleague's profile or updates to the status of a colleague. An activity type is associated with a specific activity template and is generated by a specific activity application.

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

1.2 References

1.2.1 Normative References

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

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

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

[MS-TDS] Microsoft Corporation, "Tabular Data Stream Protocol Specification", February 2008.

[MS-UPSPROF2] Microsoft Corporation, "User Profile Stored Procedures Version 2 Protocol Specification", July 2009.

[MS-WSSFO2] Microsoft Corporation, "Windows SharePoint Services (WSS): File Operations Database Communications Version 2 Protocol Specification", July 2009.

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

[XML10] World Wide Web Consortium, "Extensible Markup Language (XML) 1.0 (Third Edition)", February 2004, http://www.w3.org/TR/REC-xml

[XMLINFOSET] World Wide Web Consortium, "XML Information Set (Second Edition)", February 2004, http://www.w3.org/TR/2004/REC-xml-infoset-20040204

[XMLNS] World Wide Web Consortium, "Namespaces in XML 1.0 (Third Edition)", W3C Recommendation 8 December 2009, http://www.w3.org/TR/REC-xml-names/

[XMLSCHEMA1] Thompson, H.S., Ed., Beech, D., Ed., Maloney, M., Ed., and Mendelsohn, N., Ed., "XML Schema Part 1: Structures", W3C Recommendation, May 2001, http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/

[XMLSCHEMA2] Biron, P.V., Ed. and Malhotra, A., Ed., "XML Schema Part 2: Datatypes", W3C Recommendation, May 2001, http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/

1.2.2 Informative References

[MS-GLOS] Microsoft Corporation, "Windows Protocols Master Glossary", March 2007.

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

1.3 Protocol Overview (Synopsis)

This protocol allows clients to add or delete user profile change events from a log on the protocol server and to retrieve those user profile change events using pre-defined criteria such as events that have happened after a specified time or with a specific user.

The following figure shows an example of data flow between the protocol client and the protocol server.

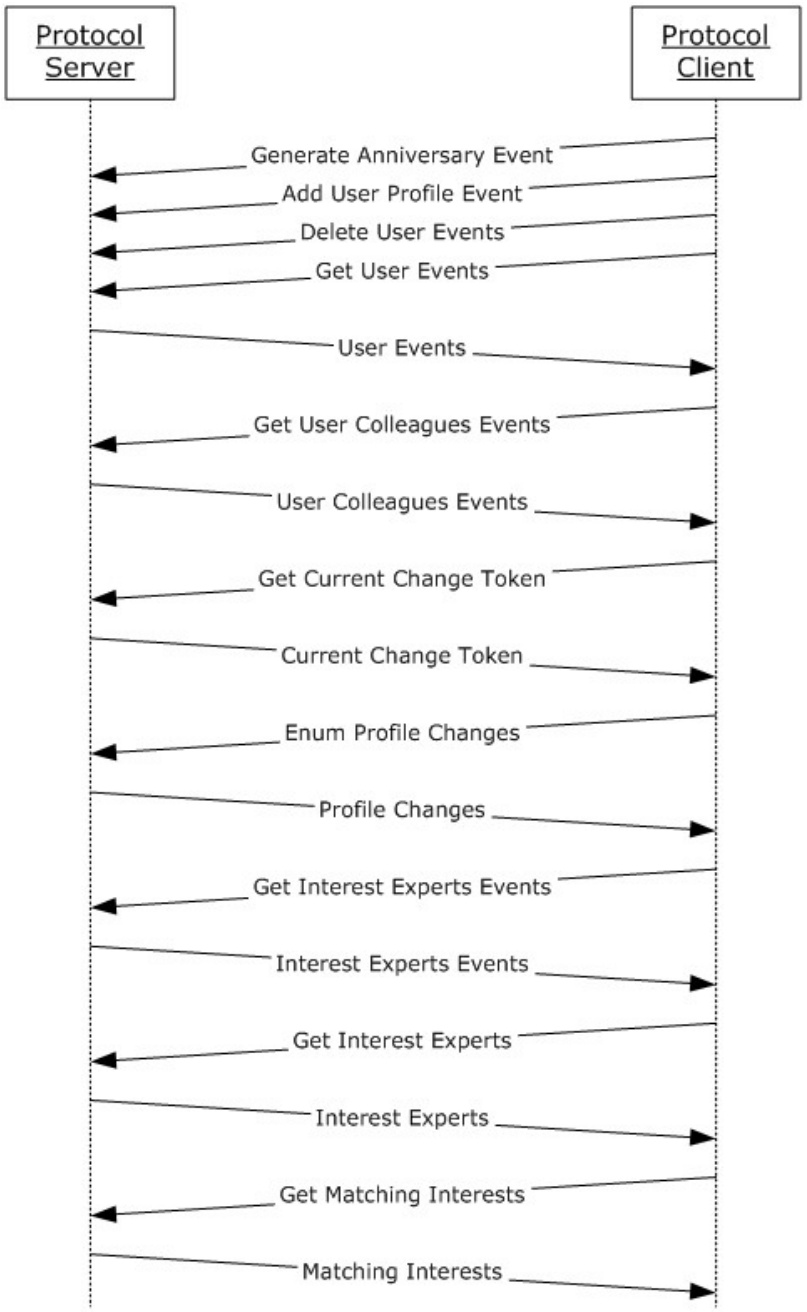

Figure 1: User Profile Change Log protocol data flow

The protocol client can ask the protocol server to generate all user profile change events with an Object Type of "Anniversary" (see section 2.2.1.3). The protocol client can ask the protocol server to add or delete user profile change events.

The protocol client can request the protocol server to provide all user profile change events for a particular user or for all of the **colleague** properties for a specified user. In another possible operation, the protocol client can request the protocol server to provide the **change token** for the last read user profile change event.

The protocol client can ask the protocol server to enumerate all user profile or **organization** profile changes.

The protocol client can ask the protocol server for users who have interests or are experts of specified interests. Similarly, the protocol client can ask the protocol server for user profile change events that contain experts or have interests that match specified interests.

This protocol provides a way for a protocol client to aggregate user profile change events and organize them into a set of activity events published by a particular user or a set of **activity events** a particular user has subscribed to.

The protocol client can ask the protocol server to retrieve activity events within a specified time interval from set of activity events published by a particular user or set of activity events a particular user has subscribed to. Every activity event has an associated type, and the protocol client can ask the protocol server to provide information about these types.

Protocol client can also use this protocol to update existing activity events or insert new activity events in the set of published activity events by a user. Similarly, protocol client can also use this protocol to update existing activity events or insert new activity events in the set of activity events subscribed by a user. In addition, protocol client can update existing activity event types, or create new activity event types using this protocol. The protocol client can use this protocol to delete activity events or event types as well.

1.4 Relationship to Other Protocols

The following diagram shows the transport stack for this protocol and the relationship to other protocols:

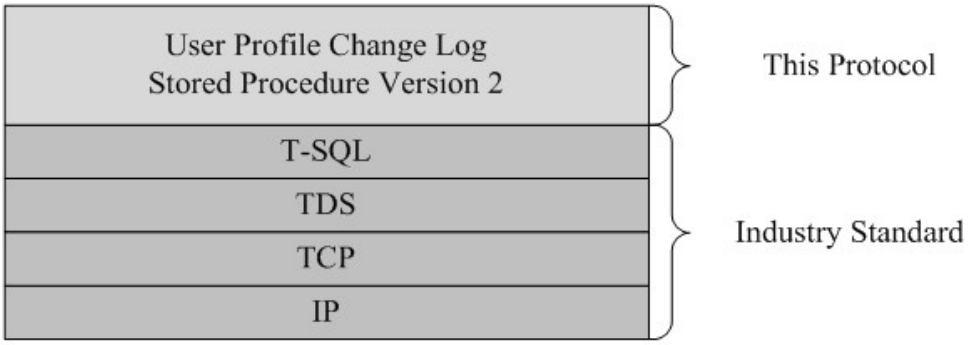

Figure 2: This protocol in relation to other protocols

1.5 Prerequisites/Preconditions

None.

1.6 Applicability Statement

This protocol is used to interact with a **change log** of user **events (2)** on protocol server.

1.7 Versioning and Capability Negotiation

Versions of the data structures or **stored procedures** in the database are required to be the same as expected by the **front-end Web server**. If the stored procedures do not provide the calling parameters or return values as expected, the results of the call are indeterminate.

The version negotiation process for this protocol is identical to the process defined in [MS-WSSFO2] section 1.7.

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2 Messages

2.1 Transport

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

2.2 Common Data Types

2.2.1 Simple Data Types and Enumerations

2.2.1.1 Privacy Policy Type

A 32-bit signed integer. Profile Privacy Policy MUST be a value from the following table.

Value	Meaning
0x1	The items that have this privacy policy setting are mandatory.
0x2	The items that have this privacy policy setting are enabled.
0x4	The items that have this privacy policy were opted to be disabled.
0x8	The items that have this privacy policy are disabled.

2.2.1.2 Change Type

A 32-bit signed integer. Change Type MUST be composed doing a BINARY OR of 1 or more values from the following table.

Value	Type	Meaning
0x1	Add	An object has been added to the profile.
0x2	Modify	An existing object has been modified.
0x4	Delete	An existing object has been deleted.
0x8	Metadata	The metadata of the object has been modified.

2.2.1.3 Object Type

A 32-bit signed integer.

For stored procedure input parameters, the Object Type value MUST be a bitmask of values from the following table composed by using a binary OR of one or more values from the following table.

For stored procedure result sets, the Object Type value MUST be one and only one of the values in the following table.

Value	Type	Meaning
0x1	SingleValueProperty	A single-value property change for a user profile.

Value	Type	Meaning
0x2	MultiValueProperty	A multivalue property change for a user profile.
0x4	Anniversary	An anniversary property change for a user profile.
0x8	DLMembership	A **distribution list membership** change for a user profile.
0x10	SiteMembership	A **site** membership change.
0x20	QuickLink	A link item change.
0x40	Colleague	Colleague relationship addition, deletion or update.
0x80	PersonalizationSite	Personalization site item change.
0x100	UserProfile	User profile change.
0x200	WebLog	User profile blog post change.
0x400	Custom	Custom **HTML** used in News Feed change.
0X800	OrganizationProfile	Organization profile change.
0x1000	OrganizationMembership	Organization membership change.

2.2.1.4 Value

The value of a user profile change entry defined by the Object Type (Section 2.2.1.3) and MUST be one of the following representations:

Object Type	Value Representation
Anniversary	**T-SQL** nvarchar value containing the anniversary **UTC** date and time, and uses the following format: "yyyy-MM-dd HH:mm:ss".
DLMembership	T-SQL nvarchar value that uniquely identifies the distribution list.
SiteMembership	T-SQL nvarchar value containing the site **GUID**.
QuickLink	T-SQL nvarchar value containing the **Uniform Resource Locator (URL)** for the link.
Colleague	T-SQL nvarchar value containing the user name of the colleague.
PersonalizationSite	T-SQL nvarchar value containing a link for the specified personal site.
UserProfile	T-SQL nvarchar value containing the name of the user whose user profile was changed.
WebLog	MUST be NULL.
SingleValueProperty or MultiValueProperty	Any of the following, depending on the property type: ▪ Boolean value represented as a string containing 0 or 1. ▪ T-SQL nvarchar value. ▪ UTC Date – encoded as "yyyy-MM-dd HH:mm:ss".

Object Type	Value Representation
	▪ Float –in single precision SQL-defined floating point number encoded as a string in the current culture. ▪ int32 – 32 bit signed integer. ▪ int64 – 64 bit signed integer.
Custom	T-SQL nvarchar value containing the custom HTML to use in News Feed.
OrganizationProfile	T-SQL nvarchar value containing the name of the organization profile.
OrganizationMembership	T-SQL nvarchar value containing the organization membership name.

2.2.1.5 Privacy Type

A 32-bit signed integer that specifies the set of users who are allowed to access a resource. The value MUST be listed in the following table:

Value	Description
1	All users are allowed to access the resource.
2	The only users allowed to access the resource are the owner of the resource and the owner's colleagues.
4	The only users allowed to access the resource are the owner of the resource and the owner's workgroup colleagues.
8	The only two users allowed to access the resource are the owner of the resource and the owner's **Manager** property.
16	The only user allowed to access the resource is the owner of the resource.

2.2.2 Bit Fields and Flag Structures

None.

2.2.3 Binary Structures

None.

2.2.4 Result Sets

2.2.4.1 GetUserColleagueEvents

The GetUserColleagueEvents result set contains the user profile change events of the Colleagues property of the specified user.

```
EventId bigint,
RecordId bigint,
ChangeType int,
EventTime datetime,
OldValue bigint,
NewValueData sql_variant,
```

```
NewValueChecksum int,
ObjectType int,
ItemSecurity int,
ChangedLinkId bigint,
ChangedColleagueId bigint,
ChangedMemberGroupId bigint,
ChangedOrganizationId bigint,
ChangedPropertyId bigint,
ChangedSourceId uniqueidentifier,
PartitionID uniqueidentifier,
```

EventId: Unique identifier for the user profile change event.

RecordId: Identifier for the user that created the user profile change event.

ChangeType: A numeric value representing the user profile Change Type.

EventTime: Date and time when the user profile change event occurred.

OldValue: MUST be NULL.

NewValueData: The current value.

NewValueChecksum: A numeric value that is the **checksum** of the actual value as returned by the T-SQL CHECKSUM function.

ObjectType: The Object Type of the user profile change event.

ItemSecurity: The Privacy Policy Type value. This value MUST NOT be NULL if Object Type is either of the values QuickLink or PersonalizationSite.

ChangedLinkId: A unique identifier for the changed link. This value MUST NOT be NULL if the Object Type is either of the values QuickLink or PersonalizationSite.

ChangedColleagueId: A record identifier of the Colleague property that has changed. This value MUST NOT be NULL if the Object Type is the value Colleague.

ChangedMemberGroupId: A record identifier of a Member Group which corresponds to the user's membership which changed. This value MUST NOT be NULL if the Object Type is either of the values DLMembership or SiteMembership.

ChangedOrganizationId: A record identifier of the Organization property that has changed. This value MUST NOT be NULL if the Object Type is the value OrganizationMembership.

ChangedPropertyId: The unique identifier for user profile property that has changed. This value MUST NOT be NULL if the Object Type is either of the values SingleValueProperty or MultiValueProperty.

ChangedSourceId: Unique identifier for the Privacy Policy record that changed. This value MUST NOT be NULL for all Object Type values, with the exception of WebLog, UserProfile, and OrganizationProfile.

PartitionID: A **partition identifier** used to filter the current request. This value MUST NOT be NULL or empty.

2.2.4.2 GetUserEvents

The GetUserEvents result set MUST contain one record identifying the most recent user profile change event prior to the specified *@MinEventTime* or *@MinEventId* parameter.

```
EventTime datetime,
EventId bigint,
```

EventTime: The date and time when the user profile change event occurred.

EventId: Unique identifier for the user profile change event.

2.2.4.3 GetUserEventsNoRecordId

The GetUserEventsNoRecordId result set returns the user profile change events for all existing users up to a maximum of 1000 records.

```
EventId bigint,
RecordId bigint,
ChangeType int,
EventTime datetime,
OldValue bigint,
NewValueData sql_variant,
NewValueChecksum int,
ObjectType int,
ItemSecurity int,
ChangedLinkId bigint,
ChangedColleagueId bigint,
ChangedMemberGroupId bigint,
ChangedOrganizationId bigint,
ChangedPropertyId bigint,
ChangedSourceId uniqueidentifier,
PartitionID uniqueidentifier,
UserID uniqueidentifier,
NTName nvarchar(400),
Email nvarchar(256),
SipAddress nvarchar(250),
ProfileSubtypeID int,
PictureUrl ntext,
PreferredName nvarchar(256),
```

EventId: Unique identifier for the user profile change event.

RecordId: Identifier for the user that created the user profile change event.

ChangeType: A numeric value representing the user profile Change Type.

EventTime: Date and time when the user profile change event occurred.

OldValue: MUST be NULL.

NewValueData: The current value.

NewValueChecksum: A numeric value that is the checksum of the actual value as returned by the T-SQL CHECKSUM function.

ObjectType: The Object Type of the user profile change event.

ItemSecurity: The Privacy Policy Type value. This value MUST NOT be NULL if the Object Type is either of the values QuickLink or PersonalizationSite.

ChangedLinkId: Unique identifier for the changed link. This value MUST NOT be NULL if the Object Type is either of the values QuickLink or PersonalizationSite.

ChangedColleagueId: A record identifier of the Colleague property that has changed. This value MUST NOT be NULL if the Object Type is the value Colleague.

ChangedMemberGroupId: A record identifier of a Member Group which corresponds to the user's membership which changed. This value MUST NOT be NULL if the Object Type is either of the values DLMembership or SiteMembership.

ChangedOrganizationId: A record identifier of the Organization property that has changed. This value MUST NOT be NULL if the Object Type is the value OrganizationMembership.

ChangedPropertyId: The unique identifier for user profile property that has changed. This value MUST NOT be NULL if the Object Type is either of the values SingleValueProperty or MultiValueProperty.

ChangedSourceId: Unique identifier for the Privacy Policy record that changed. This value MUST NOT be NULL for all Object Type values, with the exception of WebLog, UserProfile, and OrganizationProfile.

PartitionID: A partition identifier used to filter the current request. This value MUST NOT be NULL or empty.

UserID: Record identifier for the user with whom the event is associated.

NTName: User name for the user with whom the event is associated.

Email: An e-mail address for the user with whom the event is associated.

SipAddress: The **Session Initiation Protocol (SIP)** address for the user with whom the event is associated.

ProfileSubtypeID: A profile subtype identifier.

PictureUrl: URL of the picture for the user with whom the event is associated.

PreferredName: Display name for the user with whom the event is associated.

2.2.4.4 profile_EnumProfileChanges.profile_EnumProfileChanges.Default.ResultSet0

The EnumProfileChangesUser result set contains the user profile changes of the specified user.

```
EventId bigint,
EventTime datetime,
ChangedPropertyId bigint,
ChangeType int,
ObjectType int,
NewValueData sql_variant,
DN nvarchar(2048),
RecordId bigint,
```

EventId: Unique identifier for the user profile change event.

EventTime: Date and time when the user profile change event occurred.

ChangedPropertyId: Identifier of the changed property. It is one of the property identifiers passed in the @propertyids input parameter.

ChangeType: A numeric value representing the user profile Change Type.

ObjectType: The Object Type of the user profile change event.

NewValueData: The current value.

DN: **Distinguished name (DN)** of the Object Type.

RecordId: Identifier of the changed object.

2.2.4.5 GetCurrentChangeTokenUser

The GetCurrentChangeTokenUser result set contains the user profile Change Token of the most recent user profile change event.

```
EventTime datetime,
EventId bigint,
```

EventTime: Date and time when the user profile change event occurred.

EventId: Unique identifier for the user profile change event.

2.2.4.6 GetInterestsExpertsEvents

The GetInterestExpertsEvents result set contains user profile change events that match specified interests.

```
EventId bigint,
ObjectType int,
ChangedSourceId uniqueidentifier,
ChangedPropertyId bigint,
RecordId bigint,
ChangeType int,
EventTime datetime,
NewValueData sql_variant,
NTName nvarchar(400),
```

EventId: Unique identifier for the user profile change event.

ObjectType: The Object Type of the user profile change event.

ChangedSourceId: Unique identifier for the Privacy Policy record that changed. This value MUST NOT be NULL for all Object Type values, with the exception of WebLog, UserProfile, and OrganizationProfile.

ChangedPropertyId: The unique identifier for user profile property that has changed. This value MUST NOT be NULL if the Object Type is either of the values SingleValueProperty or MultiValueProperty.

RecordId: Identifier for the user that created the user profile change event.

ChangeType: A numeric value representing the user profile Change Type.

EventTime: Date and time when the user profile change event occurred.

NewValueData: The current value.

NTName: **Domain** user account name.

2.2.4.7 GetUserEventsRecordId

The GetUserEventsRecordId result set contains the user profile change events for the specified user up to a maximum of 1000 records.

```
EventId bigint,
RecordId bigint,
ChangeType int,
EventTime datetime,
OldValue bigint,
NewValueData sql_variant,
NewValueChecksum int,
ObjectType int,
ItemSecurity int,
ChangedLinkId bigint,
ChangedColleagueId bigint,
ChangedMemberGroupId bigint,
ChangedOrganizationId bigint,
ChangedPropertyId bigint,
ChangedSourceId uniqueidentifier,
PartitionID uniqueidentifier,
```

EventId: Unique identifier for the user profile change event.

RecordId: Identifier for the user that created the user profile change event.

ChangeType: A numeric value representing the user profile Change Type.

EventTime: Date and time when the user profile change event occurred.

OldValue: MUST be NULL.

NewValueData: The current value.

NewValueChecksum: A numeric value that is the checksum of the actual value as returned by the T-SQL CHECKSUM function.

ObjectType: The Object Type of the user profile change event.

ItemSecurity: The Privacy Policy Type value. This value MUST NOT be NULL if the Object Type is either of the values QuickLink or PersonalizationSite.

ChangedLinkId: Unique identifier for the changed link. This value MUST NOT be NULL if the Object Type is either of the values QuickLink or PersonalizationSite.

ChangedColleagueId: The record identifier of the Colleague property that has changed. This value MUST NOT be NULL if the Object Type is the value Colleague.

ChangedMemberGroupId: The record identifier of a Member Group which corresponds to the user's membership which changed. This value MUST NOT be NULL if the Object Type is either of the values DLMembership or SiteMembership.

ChangedOrganizationId: A record identifier of the Organization property that has changed. This value MUST NOT be NULL if the Object Type is the value OrganizationMembership.

ChangedPropertyId: The unique identifier for user profile property that has changed. This value MUST NOT be NULL if the Object Type is either of the values SingleValueProperty or MultiValueProperty.

ChangedSourceId: Unique identifier for the Privacy Policy record that changed. This value MUST NOT be NULL for all Object Type values, with the exception of WebLog, UserProfile, and OrganizationProfile.

PartitionID: A partition identifier used to filter the current request. This value MUST NOT be NULL or empty.

2.2.4.8 profile_EnumProfileChanges.profile_EnumProfileChanges.Default1.ResultSet0

The EnumProfileChangesOrganization result set contains the organization profile changes of the specified user.

```
EventId bigint,
EventTime datetime,
ChangedPropertyId bigint,
ChangeType int,
ObjectType int,
NewValueData sql_variant,
DN nvarchar(2048),
RecordId bigint,
```

EventId: Unique identifier for the organization profile change event.

EventTime: Date and time when the organization profile change event occurred.

ChangedPropertyId: Identifier of the changed property. It is one of the property identifiers passed in the @propertyids input parameter.

ChangeType: A numeric value representing the organization profile Change Type.

ObjectType: The Object Type of the organization profile change event.

NewValueData: The current value.

DN: Distinguished name (DN) of the Object Type.

RecordId: Identifier of the changed object.

2.2.4.9 GetCurrentChangeTokenOrganization

The GetCurrentChangeTokenOrganization result set contains the organization profile Change Token of the most recent organization profile change event.

```
EventTime datetime,
```

```
EventId bigint,
```

EventTime: Date and time when the organization profile change event occurred.

EventId: Unique identifier for the organization profile change event.

2.2.4.10 GetActivityApplications

The GetActivityApplications result set contains **activity application**s in descending order by the LastUpdateTime.

```
PartitionId uniqueidentifier,
ApplicationId bigint,
ApplicationName nvarchar(256),
Description nvarchar(1000),
CreationTime datetime,
LastUpdateTime datetime,
CreatedBy nvarchar(256),
ModifiedBy nvarchar(256),
```

PartitionId: A partition identifier used to filter the current request. This value MUST NOT be NULL or empty.

ApplicationId: The unique identifier of the activity application. This value MUST NOT be NULL.

ApplicationName: The name of the activity application. This value MUST NOT be NULL or empty.

Description: The description of the activity application. This value MUST NOT be NULL.

CreationTime: The datetime when the activity application was created. This value MUST NOT be NULL.

LastUpdateTime: The datetime when the activity application was last updated. This value MUST NOT be NULL. If the activity application has not been updated since creation, this value MUST be equal to the CreationTime.

CreatedBy: A string specifying the creator of the activity application.

ModifiedBy: A string specifying the last modifier of the activity application.

2.2.4.11 GetActivityEvents

The GetActivityEvents result set contains activity events stored on the server, in descending order by the LastUpdateTime.

```
PartitionId uniqueidentifier,
ActivityEventId bigint,
EntityTypeId nvarchar(50),
EntityId bigint,
ActivityTypeId bigint,
CreationTime datetime,
LastUpdateTime datetime,
TemplateVariable nvarchar(4000),
IsRolledUp bit,
```

PartitionId: A partition identifier used to filter the current request. This value MUST NOT be NULL or empty.

ActivityEventId: The unique identifier of the activity event. This value MUST NOT be NULL.

EntityTypeId: A string for the entity type. This value MUST be set to the string 'user'.

EntityId: The unique identifier of the entity that owns or subscribes to the activity event. This value MUST NOT be NULL.

ActivityTypeId: The unique identifier of the **activity type**. This value MUST NOT be NULL.

CreationTime: The creation time of the activity event. This value MUST NOT be NULL.

LastUpdateTime: The time of occurrence of the change event (2). This value MUST NOT be NULL.

TemplateVariable: A string specifying the details of the activity event. This value MUST conform to the ActivityTemplateVariable schema defined in section 2.2.6.4.1.

IsRolledUp: A flag indicating whether the activity event is a roll-up of multiple activity events of the same activity type. This value MUST be 1 if the activity is a roll-up; otherwise it MUST be 0.

2.2.4.12 GetActivityPreferences

The GetActivityPreferences result set contains a list of unique identifiers of those activity types that the specified user does not subscribe to activity feed events for.

```
PartitionId uniqueidentifier,
EntityId bigint,
ActivityTypeId bigint,
```

PartitionId: A GUID used to filter the current request. This value MUST NOT be NULL or empty.

EntityId: The unique identifier of the user who doesn't subscribe to activity feed events of the activity type specified by the ActivityTypeId.

ActivityTypeId: The unique identifier of the activity type.

2.2.4.13 GetActivityTemplates

The GetActivityTemplates result set contains **activity template**s in descending order by the LastUpdateTime.

```
PartitionId uniqueidentifier,
ActivityTemplateId bigint,
ActivityTypeId bigint,
CreationTime datetime,
LastUpdateTime datetime,
TitleFormatLocStringName nvarchar(256),
TitleFormatLocStringResourceFile nvarchar(256),
IsMultivalued bit,
CreatedBy nvarchar(256),
ModifiedBy nvarchar(256),
```

PartitionId: A partition identifier used to filter the current request. This value MUST NOT be NULL or empty.

ActivityTemplateId: The unique identifier of the activity template. This value MUST NOT be NULL.

ActivityTypeId: The unique identifier of the activity type. This value MUST NOT be NULL.

CreationTime: The datetime when the activity template was created. This value MUST NOT be NULL.

LastUpdateTime: The datetime when the activity template was last updated. This value MUST NOT be NULL. If the activity template has not been updated since creation, this value MUST be equal to the CreationTime.

TitleFormatLocStringName: The key string for looking up (in the title format string resource file) the localized template string to be used when generating a human-readable description of an event of the associated activity type. This value MUST NOT be NULL or empty.

TitleFormatLocStringResourceFile: The name of the resource file containing the localized template string to be used when generating a human-readable description of an event of the associated activity type. This value MUST NOT be NULL or empty.

IsMultivalued: The IsMultivalued attribute of the activity template. This value MUST be 0 if the template takes a single value; otherwise (that is, it takes a list of values) it MUST be 1.

CreatedBy: The creator of the activity template.

ModifiedBy: The last modifier of the activity template.

2.2.4.14 GetActivityTypes

The GetActivityTypes result set contains activity types in descending order by the LastUpdateTime.

```
PartitionId uniqueidentifier,
ActivityTypeId bigint,
ApplicationId bigint,
ActivityTypeName nvarchar(256),
ActivityTypeNameLocStringResourceFile nvarchar(256),
ActivityTypeNameLocStringName nvarchar(256),
Description nvarchar(1000),
CreationTime datetime,
LastUpdateTime datetime,
AllowRollup bit,
IsConsolidated bit,
IsPublished bit,
GatheringInterval int,
ItemTimeToLive int,
CreatedBy nvarchar(256),
ModifiedBy nvarchar(256),
```

PartitionId: A partition identifier used to filter the current request. This value MUST NOT be NULL or empty.

ActivityTypeId: The unique identifier of the activity type. This value MUST NOT be NULL.

ApplicationId: The unique identifier of the ApplicationId. This value MUST NOT be NULL.

ActivityTypeName: The name of the activity type. This value MUST NOT be NULL or empty.

ActivityTypeNameLocStringResourceFile: The name of the resource file containing the localized activity type name. This value MUST NOT be NULL or empty.

ActivityTypeNameLocStringName: The key string for looking up the localized activity type name in the activity type name localized string resource file. This value MUST NOT be NULL or empty.

Description: A string representing the description of the activity type.

CreationTime: The datetime when the activity type was created. This value MUST NOT be NULL.

LastUpdateTime: The datetime when the activity type was last updated. This value MUST NOT be NULL or empty. If the activity type has not been updated since creation, this value MUST be equal to the CreationTime.

AllowRollup: A flag indicating whether activity events of this activity type are allowed to be rolled up (section 3.1.1) in GetActivityEvents result sets (section 2.2.4.11). This value MUST be 1 if the roll-ups are allowed; otherwise it MUST be 0.

IsConsolidated: A flag indicating whether activity events of this type are to be pre-consolidated for subscribing entities, so that they can be immediately consumed by applications. If so, this value MUST be 1; otherwise it MUST be 0.

IsPublished: Reserved. This value MUST be 1.

GatheringInterval: Reserved. This value MUST be 3600.

ItemTimeToLive: Reserved. This value MUST be 84600.

CreatedBy: The creator of the activity type.

ModifiedBy: The last modifier of the activity type.

2.2.4.15 GetLatestLastActivityEventUpdateTime

The GetLatestLastActivityEventUpdateTime result set contains the most recent update time from among the activity events meeting specified criteria.

```
LastUpdateTime datetime,
```

LastUpdateTime: The most recent update time from among the specified activity events.

2.2.4.16 GetUserInfo

The GetUserInfo result set contains information about selected users.

```
RecordId bigint,
Email nvarchar(256),
PreferredName nvarchar(256),
NTName nvarchar(400),
PictureUrl nvarchar(max),
```

RecordId: The unique identifier of the user profile. This value MUST NOT be NULL.

Email: The user's e-mail address.

PreferredName: The user's preferred name.

NTName: The user's login. This value MUST NOT be NULL or empty.

PictureUrl: The URL of the user's picture.

2.2.4.17 GetUsersColleaguesAndRights

The GetUsersColleaguesAndRights result set contains information about selected users, those users' colleagues, and the visibility of each colleague's events to the associated user.

```
RecordId bigint,
Email nvarchar(256),
PreferredName nvarchar(256),
NTName nvarchar(400),
PictureUrl nvarchar(max),
ColleagueRecordId bigint,
ItemSecurity int,
UnnamedColumn7 int,
```

RecordId: The unique identifier of the user's user profile. This value MUST NOT be NULL.

Email: The user's e-mail address.

PreferredName: The user's preferred name.

NTName: The user's login. This value MUST NOT be NULL or empty.

PictureUrl: The URL of the user's picture.

ColleagueRecordId: The unique identifier of the user profile of the user's colleague. This value MUST NOT be NULL.

ItemSecurity: MUST be a Privacy Type (section 2.2.1.5) value that defines privacy type of the user's colleague.

UnnamedColumn7: MUST be a Privacy Type (section 2.2.1.5) value that defines the visibility of the colleague's events to the user.

2.2.4.18 GetUserEventsIDOldest

The GetUserEventsIDOldest result set contains the unique identifier of the oldest activity event that was last updated after a specified datetime.

```
EventID bigint,
```

EventID: The unique identifier of the activity event. This value MUST NOT be NULL.

2.2.4.19 GetUserEventsIDNewest

The GetUserEventsIDNewest result set contains the unique identifier of the latest activity event that was last updated after a specified datetime.

```
EventID bigint,
```

EventID: The unique identifier of the activity event. This value MUST NOT be NULL.

2.2.5 Tables and Views

None.

2.2.6 XML Structures

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

2.2.6.1 Namespaces

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

Prefix	Namespace URI	Reference
s	http://www.w3.org/2001/XMLSchema	[XMLSCHEMA1] [XMLSCHEMA2]
tns	AF	

2.2.6.2 Simple Types

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

2.2.6.3 Complex Types

The following table summarizes the set of common XML Schema complex type definitions defined by this specification. XML Schema complex type definitions that are specific to a particular operation are described with the operation.

Complex Type	Description
Link	A URI associated with syndication content.
List	A list of syndication content links.
Entity	The owner or publisher of a syndication.

2.2.6.3.1 Link

Target namespace: AF

Referenced by: List (section 2.2.6.3.2) , ActivityTemplateVariable (section 2.2.6.4.1).

This type specifies a syndication content Link.

Child Elements:

Name : A string (as specified in [XMLSCHEMA2]) element that specifies the name of the link.

Value : A string (as specified in [XMLSCHEMA2]) element that specifies the value of the link.

Href : An anyURI (as specified in [XMLSCHEMA2]) element that specifies the href of the link.

```
<s:complexType name="Link">
  <s:sequence>
    <s:element name="Name" type="s:string" minOccurs="0" maxOccurs="1" />
    <s:element name="Value" type="s:string" minOccurs="0" maxOccurs="1" />
    <s:element name="Href" type="s:anyURI" minOccurs="0" maxOccurs="1" />
  </s:sequence>
</s:complexType>
```

2.2.6.3.2 List

Target namespace: AF

Referenced by: ActivityTemplateVariable (section 2.2.6.4.1).

This type specifies a list of Link (section 2.2.6.3.1) elements.

Child Elements:

Size : A long (as specified in [XMLSCHEMA2]) element that specifies the number of items in the list.

Separator : A string (as specified in [XMLSCHEMA2]) element that specifies the item separator for the list.

Items : A sequence of Link elements that specifies the Link items in the list.

```
<s:complexType name="List">
  <s:sequence>
    <s:element name="Size" type="s:long" minOccurs="1" maxOccurs ="1"/>
    <s:element name="Separator" type="s:string" minOccurs="0" maxOccurs ="1"/>
    <s:element name="Items" type="tns:Link" minOccurs="0" maxOccurs="unbounded"/>
  </s:sequence>
</s:complexType>
```

2.2.6.3.3 Entity

Target namespace: AF

Referenced by: ActivityTemplateVariable (section 2.2.6.4.1).

This type specifies the owner or publisher of a syndication.

Child Elements:

Email : A string (as specified in [XMLSCHEMA2]) element that specifies e-mail address of the entity.

Type : A string (as specified in [XMLSCHEMA2]) element that specifies the type of the entity (user, organization, group, contact, and so on).

Name : A string (as specified in [XMLSCHEMA2]) element that specifies the name of the entity.

Href : An anyURI (as specified in [XMLSCHEMA2]) element that specifies an href for the entity.

Picture : An anyURI (as specified in [XMLSCHEMA2]) element that specifies the address of a picture for the entity.

Id : A long (as specified in [XMLSCHEMA2]) that specifies the unique entity ID of the entity.

```
<s:complexType name="Entity">
  <s:sequence>
    <s:element name="Email" type="s:string" minOccurs="0" maxOccurs="1" />
    <s:element name="Type" type="s:string" minOccurs="0" maxOccurs ="1" />
    <s:element name="Name" type="s:string" minOccurs="0" maxOccurs="1" />
    <s:element name="Href" type="s:anyURI" minOccurs="0" maxOccurs="1" />
    <s:element name="Picture" type="s:anyURI" minOccurs="0" maxOccurs="1" />
    <s:element name="Id" type="s:long" minOccurs="1" maxOccurs="1" />
  </s:sequence>
</s:complexType>
```

2.2.6.4 Elements

The following table summarizes the set of common XML Schema element definitions defined by this specification. XML Schema element definitions that are specific to a particular operation are described with the operation.

Element	Description
ActivityTemplateVariable	Properties of a syndication item.

2.2.6.4.1 ActivityTemplateVariable

Target namespace: AF

This element specifies properties of a syndication item.

Child Elements:

Owner : An Entity (as specified in section 2.2.6.3.3) element that specifies the owner of the syndication.

Publisher : An Entity (as specified in section 2.2.6.3.3) element that specifies the publisher of the syndication.

PublishDate : A dateTime (as specified in [XMLSCHEMA2]) element that specifies the most recent publication date of the syndication.

Name : A string (as specified in [XMLSCHEMA2]) element that specifies the name of the syndication.

Value : A string (as specified in [XMLSCHEMA2]) element that specifies the value associated with the syndication name.

Link : An Link (as specified in section 2.2.6.3.1) element that specifies base URI for the syndication.

DateOnly : A date (as specified in [XMLSCHEMA2]) element for birthdays, anniversaries, and other events where time is not significant.

List : A List (as specified in section 2.2.6.3.2) element that specifies a list of syndication content links.

2.2.6.5 Attributes

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

2.2.6.6 Groups

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

2.2.6.7 Attribute Groups

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

3 Protocol Details

3.1 Server Details

3.1.1 Abstract Data Model

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

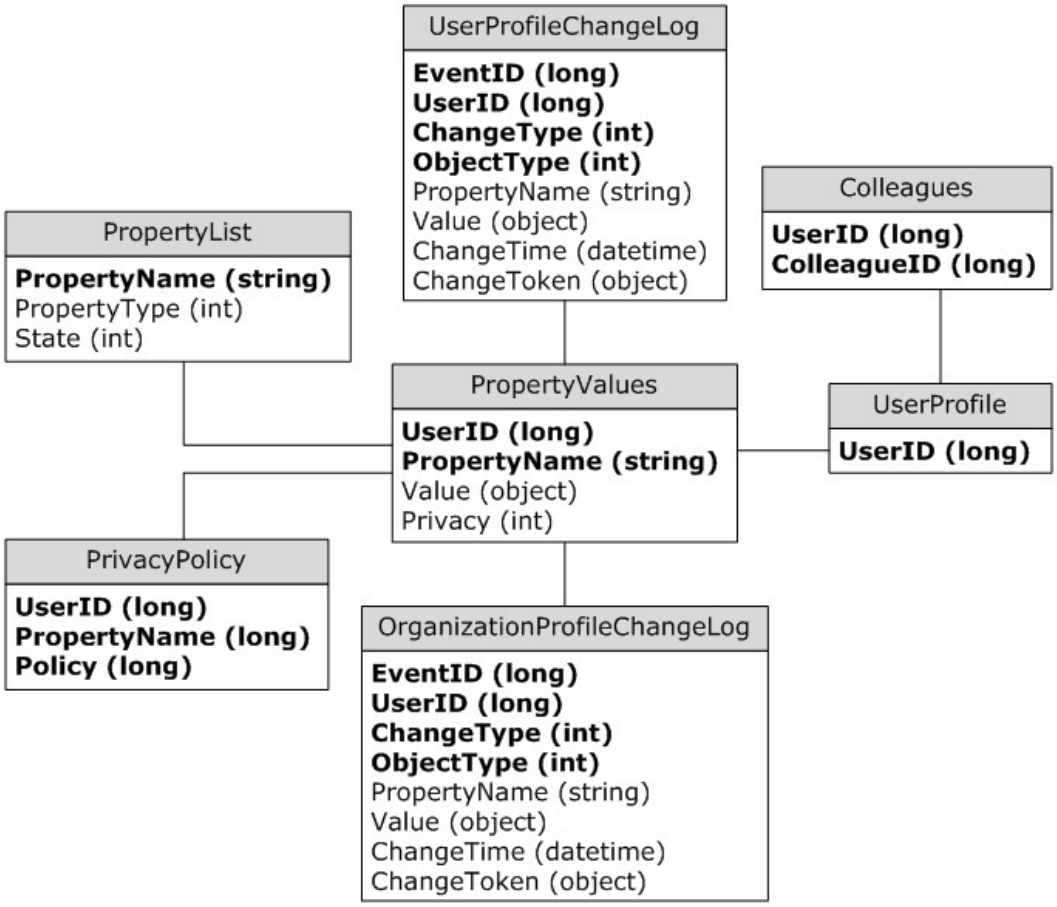

Figure 3: Abstract data model

In the preceding figure, each table specifies a type of entity in the model, and each line specifies that one type of entity always contains a reference to another.

UserProfile Table: A collection of entries corresponding to the user profile available on the server.

- **UserID:** A unique identifier for a user.

Colleagues Table: A collection of entries, where each one corresponds to a colleague of a user.

- **UserID:** The unique identifier of the user.
- **ColleagueID:** The unique identifier of the user colleague.

PropertyList Table: A collection of entries, each one corresponding to a property.

- **PropertyName:** A unique name for the property.
- **PropertyType:** An integer identifying the type of the property.
- **State:** An integer identifying what is the property state.

PropertyValues Table: A collection of entries with the values of the user properties.

- **UserID:** The unique identifier of the user.
- **PropertyName:** The unique name of the property.
- **Value:** The value of the property identified by **PropertyName** for the user identified by **UserID**.
- **Privacy:** An integer indicating what is the privacy policy for this property value.

UserProfileChangeLog Table: A collection of entries containing the user profile change entries.

- **EventID:** A unique identifier for the user profile change log entry.
- **UserID:** The unique identifier for the user for whom the user profile change log entry was created.
- **ChangeType:** An integer identifying the Change Type of the user profile change log entry.
- **ObjectType:** An integer identifying the Object Type of the user profile change log entry.
- **PropertyName:** The name of the single-value property or multivalue property which the user profile change log entry refers to.
- **Value:** The value of the property for the user profile change log entry.
- **ChangeTime:** The Date and Time the user profile change log entry was created.
- **ChangeToken:** The Change Token of the user profile change log entry.

OrganizationProfileChangeLog Table: A collection of entries containing the organization profile change entries.

- **EventID:** A unique identifier for the organization profile change log entry.
- **UserID:** The unique identifier for the organization for whom the organization profile change log entry was created.
- **ChangeType:** An integer identifying the Change Type of the organization profile change log entry.
- **ObjectType:** An integer identifying the Object Type of the organization profile change log entry.
- **PropertyName:** The name of the single-value property or multivalue property which the organization profile change log entry refers to.
- **Value:** The value of the property for the organization profile change log entry.

- **ChangeTime:** The Date and Time the organization profile change log entry was created.
- **ChangeToken:** The Change Token of the organization profile change log entry.

PrivacyPolicy Table: A collection of entries containing the privacy policy of a user.

- **UserID:** The unique identifier of the user.
- **PropertyID:** An integer specifying the property field.
- **Policy:** An integer specifying the privacy of the property identified by **PropertyID** for the user identified by **UserID**.

Some additional conceptual models of data organization maintained by an implementation to participate in this protocol are explained in the following:

ActivityEvent: A construct representing an activity event. It holds information about who generated the event, what the event was, when the event happened, and what is the type of the activity event.

Application: An implementation of the protocol that adds activity events to the store on the protocol server. Each application can add activity events of different types, but each type is used by only one application.

ActivityType: A construct that represents type of an activity event.

Activity Template: A construct that defines syntax for representation of data contained in an activity event in a specific locale. There is a separate activity template for each locale supported by protocol server. A protocol server stores a set of activity templates which define syntax for the same type of activity event in all locales supported by protocol server.

Roll-up: If multiple activity events of the same type occur within a specified time interval; they can be stored as a single activity event on the server after aggregating all the event data into the single roll-up event.

3.1.2 Timers

None.

3.1.3 Initialization

None.

3.1.4 Higher-Layer Triggered Events

None.

3.1.5 Message Processing Events and Sequencing Rules

None.

3.1.5.1 profile_GetUserEvents

The profile_GetUserEvents stored procedure is called to retrieve user profile change events.

```
PROCEDURE profile_GetUserEvents (
```

```
,@RecordId bigint = NULL
@partitionID uniqueidentifier
@ViewerRights int
,@MinEventId bigint = NULL
,@MinEventTime datetime = NULL
,@ChangeTypeMask int
,@ObjectTypeMask int
,@SortDescending int = 0
,@correlationId uniqueidentifier = null
);
```

@RecordId: The record identifier of a user, or NULL. If RecordId is specified, profile_GetUserEvents MUST return the user profile change events for the user identified. If RecordId is NULL, then profile_GetUserEvents MUST return all available user profile change events for existing users.

@partitionID: A partition identifier used to filter the current request. This value MUST NOT be NULL or empty.

@ViewerRights: A bitmask specifying the privacy policy type that MUST be satisfied by the returned user profile change events. Each bit corresponds to a privacy level. If RecordId is not NULL, then this value MUST be specified. For example, 0x0C signifies the Manager and Organization Privacy Policy Type values. This parameter SHOULD be at least 0x01 (Public).

Value	Description
1	All users are allowed to access the resource.
2	The only users allowed to access the resource are the owner of the resource and the owner's colleagues.
4	The only users allowed to access the resource are the owner of the resource and the owner's workgroup colleagues.
8	The only two users allowed to access the resource are the owner of the resource and the manager of the owner.
16	The only user allowed to access the resource is the owner of the resource.

@MinEventId: A unique identifier for user profile change events. If @MinEventId is not NULL then all user profile change events with identifiers greater than this value MUST be returned. If @MinEventId is NULL then @MinEventTime MUST NOT be NULL.

@MinEventTime: A value representing the date and time for which all returned user profile change events MUST be more recent. If NULL then profile_GetUserEvents MUST return all available user profile change events. If @MinEventId is NULL then @MinEventTime MUST NOT be NULL. If @MinEventId is specified then @MinEventTime MUST be ignored.

@ChangeTypeMask: A Change Type value that specifies the user profile Change Types of the user profile change events that MUST be returned.

@ObjectTypeMask: An Object Type value that specifies the Object Type of the user profile change events that MUST be returned.

@SortDescending: An integer that specifies the sort order of the result set. If @SortDescending is 1, then the result set MUST be returned in descending order.

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

Return Values: An integer, which MUST be 0.

Result Sets:

If @MinEventId and @MinEventTime are both NULL, then profile_GetUserEvents MUST not return any result sets.

For the following combination of parameters,

@MinEventId: MUST not be NULL

@MinEventTime: MUST not be NULL

This stored procedure MUST return a GetUserEvents

This stored procedure MUST also return GetUserEventsNoRecordId or GetUserEventsRecordId. It returns GetUserEventsNoRecordId if @RecordId is NULL; otherwise it returns GetUserEventsRecordId. If @SortDescending is 1, then the result set MUST be sorted in descending order.

3.1.5.2 profile_DeleteUserEvents

The profile_DeleteUserEvents stored procedure is called to delete all user profile change events older than a given date and time.

```
PROCEDURE profile_DeleteUserEvents (
,@MinEventTime datetime
@partitionID uniqueidentifier
,@correlationId uniqueidentifier = null
);
```

@MinEventTime: A **datetime** value which specifies the date and time prior to which all user profile change events MUST be removed from the database table. This parameter MUST NOT be NULL.

@partitionID: A partition identifier used to filter the current request. This value MUST NOT be NULL or empty.

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

Return Values: An integer which MUST be the number of user profile change events that were removed from the database table.

Result Sets: MUST NOT return any result sets.

3.1.5.3 profile_GetUserColleagueEvents

The profile_GetUserColleagueEvents stored procedure is called to retrieve user profile change events for all Colleague properties of a specified user occurring after @MinEventTime.

```
PROCEDURE profile_GetUserColleagueEvents (
,@RecordId bigint
@partitionID uniqueidentifier
@MinEventTime datetime
,@ChangeTypeMask int
,@ObjectTypeMask int
,@correlationId uniqueidentifier = null
```

```
);
```

@RecordId: The record identifier of a user, which MUST not be NULL.

@partitionID: A partition identifier used to filter the current request. This value MUST NOT be NULL or empty.

@MinEventTime: A datetime value that specifies the date and time that each of the returned user profile change events MUST be more recent. If NULL, then profile_GetUserColleagueEvents MUST return all available Colleague property-related user profile change events.

@ChangeTypeMask: A Change Type value that specifies the user profile Change Types of the user profile change events that MUST be returned.

@ObjectTypeMask: An Object Type value that specifies the Object Type of the user profile change events that MUST be returned.

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

Return Values: An integer, which MUST be 0.

Result Sets:

If the Colleague Privacy Policy Type is disabled then profile_GetuserColleagueEvents MUST not return a result set.

Otherwise, this stored procedure MUST return a GetUserColleagueEvents

3.1.5.4 profile_GenerateAnniversaryEvents

The profile_GenerateAnniversaryEvents stored procedure is called to create a user profile change event for each upcoming Anniversary property of all date properties, unless the Anniversary Privacy Policy Type is disabled. In this case, a user profile change event MUST NOT be created. Each user profile change event that is created MUST have a user profile Change Type of "Add" and an Object Type of "Anniversary". If executed multiple times, profile_GenerateAnniversaryEvents MUST NOT create more than one user profile change event for the same anniversary. profile_GenerateAnniversaryEvents is defined using T-SQL syntax as follows:

```
PROCEDURE profile_GenerateAnniversaryEvents (
,@DaysAheadToScan int
@partitionID uniqueidentifier
,@correlationId uniqueidentifier = null
);
```

@DaysAheadToScan: Specifies the number of days, starting from the current date, to scan ahead for an anniversary. This parameter MUST be 3.

@partitionID: A partition identifier used to filter the current request. This value MUST NOT be NULL or empty.

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

Return Values: An integer, which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.1.5.5 profile_GetCurrentChangeToken

The profile_GetCurrentChangeToken stored procedure is called to retrieve the profile Change Token of the most recent profile change event. profile_GetCurrentChangeToken is defined using T-SQL syntax as follows:

```
PROCEDURE profile_GetCurrentChangeToken (
@partitionID uniqueidentifier
,@ProfileTypeID smallint
,@correlationId uniqueidentifier = null
);
```

@partitionID: A GUID used to filter the current request. This value MUST NOT be null or empty.

@ProfileTypeID: Specifies the type of profile to retrieve. This value MUST be from the following table.

Value	Description
1	Represents a User Profile
2	Represents an Organization Profile

@correlationId: This value MUST be ignored by the server.

Return Values: An integer, which MUST be 0.

Result Sets:

For the following combination of parameters,

@ProfileTypeID: 1

This stored procedure MUST return a GetCurrentChangeTokenUser

For the following combination of parameters,

@ProfileTypeID: 2

This stored procedure MUST return a GetCurrentChangeTokenOrganization

3.1.5.6 profile_AddUserProfileEvent

The profile_AddUserProfileEvent stored procedure is called to create a user profile change event.

```
PROCEDURE profile_AddUserProfileEvent (
@partitionID uniqueidentifier
,@RecordId bigint
,@ChangeType int
,@ObjectType int
,@ItemSecurity int = null
,@ChangedLinkId int = null
,@ChangedColleagueId int = null
,@ChangedMemberGroupId int = null
,@PropertyID int = null
,@ChangedSourceId uniqueidentifier = null
```

```
,@Data sql_variant
,@correlationId uniqueidentifier = null
);
```

@partitionID: A GUID used to filter the current request. This value MUST NOT be null or empty.

@RecordId: Identifier for the user that created the user profile change event.

@ChangeType: A numeric value representing the user profile Change Type.

@ObjectType: The Object Type of the user profile change event.

@ItemSecurity: The Privacy Policy Type value or NULL.

@ChangedLinkId: Unique identifier for the changed link or NULL.

@ChangedColleagueId: The record identifier of the Colleague property or NULL.

@ChangedMemberGroupId: The record identifier of a Member Group, which corresponds to the user's membership or NULL.

@PropertyID: The unique identifier for user profile property or NULL.

@ChangedSourceId: The unique identifier for the Privacy Policy record or NULL.

@Data: The current value.

@correlationId: This value MUST be ignored by the server.

Return Values: An integer, which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.1.5.7 profile_EnumProfileChanges

The profile_EnumProfileChanges stored procedure is called to enumerate all profile changes for a user or organization.

```
PROCEDURE profile_EnumProfileChanges (
@partitionID uniqueidentifier
,@objectType nvarchar(20)
,@propertyIds nvarchar(4000)
,@beginID bigint
,@pageSize int
,@correlationId uniqueidentifier = null
);
```

@partitionID: A GUID used to filter the current request. This value MUST NOT be null or empty.

@objectType: A string representing the object type for enumerating changes. It MUST be one of the following values: ‘user’ or ‘group’.

@propertyIds: A string list of change property IDs (defined in PropertyList table) separated by semicolon ";", used to filter enumerated changes. This criteria will filter out change properties that are not in this list.

@beginID: The first change identifier to begin with.

@pageSize: Maximum number of entries to be returned.

@correlationId: This value MUST be ignored by the server.

Return Values: An integer, which MUST be 0.

Result Sets:

For the following combination of parameters,

@ProfileType: 1

This stored procedure MUST return a profile_EnumProfileChanges.profile_EnumProfileChanges.Default.ResultSet0

For the following combination of parameters,

@ProfileType: 2

This stored procedure MUST return a profile_EnumProfileChanges.profile_EnumProfileChanges.Default1.ResultSet0

3.1.5.8 profile_GetInterestsExpertsEvents

The profile_GetInterestsExpertsEvents stored procedure returns user profile change events that match specified interests.

```
PROCEDURE profile_GetInterestsExpertsEvents (
@RecordId bigint = 0
,@terms nvarchar(max)
,@excludeInterests bit = 0
,@excludeExperts bit = 0
,@partitionID uniqueidentifier
,@correlationId uniqueidentifier = null
);
```

@RecordId: A record identifier of a user, or NULL. If RecordId is specified, profile_GetInterestExpertsEvents MUST return the user profile change events for the user identified. If @RecordId is NULL, then profile_ GetInterestExpertsEvents MUST return an empty result set.

@terms: List of semicolon separated GUID representing interests of a profile specified by @RecordId.

@excludeInterests: If @excludeInterests is not 0, no matching interests are returned.

@excludeExperts: If @excludeExperts is not 0, no matching experts are returned.

@partitionID: A partition identifier used to filter the current request. This value MUST NOT be NULL or empty.

@correlationId: This value MUST be ignored by the server.

Return Values: An integer, which MUST be 0.

Result Sets:

If @excludeInterests and @excludeExperts are both not 0 or the privacy policy for Interests and Experts are disabled, profile_GetInterestsExpertsEvents MUST not return a result set.

Otherwise, for the following combination of parameters,

@RecordId: MUST not be NULL

@terms: MUST not be NULL

This stored procedure MUST return a GetInterestsExpertsEvents

3.1.5.9 profile_GetUserColleagueEventsByDate

The GetUserColleagueEventsByDate stored procedure is called to retrieve the @Top most recent records of user profile change events for all Colleague properties of a specified user. If the Colleague Privacy Policy Type is disabled, then profile_GetuserColleagueEventsByDate MUST not return a result set.

```
PROCEDURE profile_GetUserColleagueEventsByDate (
@partitionID uniqueidentifier
,@RecordId bigint
,@Top int
,@ChangeTypeMask int
,@ObjectTypeMask int
,@correlationId uniqueidentifier = null
);
```

@partitionID: A partition identifier used to filter the current request. This value MUST NOT be NULL or empty.

@RecordId: The record identifier of a user. It MUST return the user profile change events for all Colleague properties of the specified user.

@Top: The number of records to return.

@ChangeTypeMask: A Change Type value that specifies the user profile Change Types of the user profile change events that MUST be returned.

@ObjectTypeMask: An Object Type value that specifies the Object Type of the user profile change events that MUST be returned.

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

Return Values: An integer, which MUST be 0.

Result Sets:

This stored procedure MUST return a GetUserColleagueEvents

3.1.5.10 activityFeed_DeleteActivityEventsConsolidated

The activityFeed_DeleteActivityEventsConsolidated stored procedure is called to remove all activity events older than the specified datetime, which were subscribed to by anyone from the server.

```
PROCEDURE activityFeed_DeleteActivityEventsConsolidated (
@partitionID uniqueidentifier
```

```
,@activityTypeID bigint = null
,@minEventTime datetime = null
,@batchSize int = null
,@correlationID uniqueidentifier
);
```

@partitionID: A partition identifier used to filter the current request. This value MUST NOT be NULL or empty.

@activityTypeID: Reserved. This value MUST be ignored by the server.

@minEventTime: A datetime specifying that activity events with last update time older than this value MUST be removed from the server. The value MUST NOT be NULL.

@batchSize: A positive integer specifying the maximum number of events to remove from the server in one transaction. The value MUST be less than or equal to 1000.

@correlationID: This value MUST be ignored by the server.

Return Values: An integer, which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.1.5.11 activityFeed_DeleteActivityEventsPublished

The activityFeed_DeleteActivityEventsPublished stored procedure is called to remove all activity events older than the specified datetime, which were published by anyone from the server.

```
PROCEDURE activityFeed_DeleteActivityEventsPublished (
@partitionID uniqueidentifier
,@activityTypeID bigint = null
,@minEventTime datetime = null
,@batchSize int = null
,@correlationID uniqueidentifier = null
);
```

@partitionID: A partition identifier used to filter the current request. This value MUST NOT be NULL or empty.

@activityTypeID: This value MUST be ignored by the server.

@minEventTime: A datetime specifying that activity events with last update time older than this value MUST be removed from the server. The value MUST NOT be NULL.

@batchSize: A positive integer specifying the maximum number of events to remove from the server in one transaction. The value MUST be less than or equal to 1000.

@correlationID: This value MUST be ignored by the server.

Return Values: An integer, which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.1.5.12 activityFeed_DeleteActivityTemplates

The activityFeed_DeleteActivityTemplates stored procedure is called to delete the specified activity template from the server. If an activity template with the specified activityTemplateID is not found on the server, the stored procedure MUST NOT make any change to the server.

```
PROCEDURE activityFeed_DeleteActivityTemplates (
@correlationID uniqueidentifier
,@partitionID uniqueidentifier
,@activityTemplateID bigint
);
```

@correlationID: This value MUST be ignored by the server.

@partitionID: A GUID used to filter the current request. This value MUST NOT be NULL or empty.

@activityTemplateID: The unique identifier of the activity template to be deleted. This value MUST NOT be NULL.

Return Values: An integer, which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.1.5.13 activityFeed_DeleteActivityTypes

The activityFeed_DeleteActivityTypes stored procedure is called to delete the specified activity type from the server. If an activity type with the specified activityTypeID is not found on the server, the stored procedure MUST NOT make any change to the server.

```
PROCEDURE activityFeed_DeleteActivityTypes (
@correlationID uniqueidentifier
,@partitionID uniqueidentifier
,@activityTypeID bigint
);
```

@correlationID: This value MUST be ignored by the server.

@partitionID: A partition identifier used to filter the current request. This value MUST NOT be NULL or empty.

@activityTypeID: The unique identifier of the activity type to be deleted. This value MUST NOT be NULL.

Return Values: An integer, which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.1.5.14 activityFeed_DeleteIfExistsActivityPreference

The activityFeed_DeleteIfExistsActivityPreference stored procedure is called to delete the exclusion of the specified activity type from the activity feed subscription of the specified user from the server. If the specified activity type is not excluded from the activity feed subscription of the specified user, the stored procedure MUST NOT make any change to the server.

```
PROCEDURE activityFeed_DeleteIfExistsActivityPreference (
```

```
@correlationID uniqueidentifier
,@partitionID uniqueidentifier
,@entityID bigint
,@activityTypeID bigint
);
```

@correlationID: This value MUST be ignored by the server.

@partitionID: A GUID used to filter the current request. This value MUST NOT be NULL or empty.

@entityID: The unique identifier of the user. This value MUST NOT be NULL.

@activityTypeID: The unique identifier of the activity type whose exclusion from the activity feed subscription of the user specified by @entityID is to be deleted. This value MUST NOT be NULL.

Return Values: An integer, which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.1.5.15 activityFeed_GetActivityApplications

The activityFeed_GetActivityApplications stored procedure is called to retrieve the activity application specified either by a unique identifier or a name. If neither the unique identifier nor the name are specified, the stored procedure retrieves all the activity applications stored on the server.

```
PROCEDURE activityFeed_GetActivityApplications (
@correlationID uniqueidentifier
,@partitionID uniqueidentifier
,@applicationID bigint = null
,@applicationName nvarchar(256) = null
);
```

@correlationID: This value MUST be ignored by the server.

@partitionID: A partition identifier used to filter the current request. This value MUST NOT be NULL or empty.

@applicationID: The unique identifier of the activity application to be retrieved. If @applicationID is not NULL, the stored procedure MUST retrieve the activity application with the specified unique identifier. If an activity application with the specified unique identifier does not exist on the server, the stored procedure MUST return an empty result set.

@applicationName: The name of the activity application to be retrieved. If @applicationID is not NULL, @applicationName MUST be ignored by the server. If @applicationID is NULL and @applicationName is not NULL, the stored procedure MUST retrieve the activity application with the specified name. If an activity application with the specified name does not exist on the server, the stored procedure MUST return an empty result set.

Return Values: An integer, which MUST be 0.

Result Sets:

This stored procedure MUST return a GetActivityApplications

If @applicationID is NULL and @applicationName is NULL, the result set MUST contain all activity applications stored on the server. If there are no activity applications stored on the server, the result set MUST be empty.

3.1.5.16 activityFeed_GetActivityEventsConsolidated

The activityFeed_GetActivityEventsConsolidated stored procedure is called to retrieve activity events generated by other entities and subscribed for viewing by the specified entity. Activity events newer than the specified datetime stored on the server are retrieved.

```
PROCEDURE activityFeed_GetActivityEventsConsolidated (
@correlationID uniqueidentifier
,@partitionID uniqueidentifier
,@entityID bigint
,@minEventTime datetime
,@batchSize int
,@useStoredPreferences bit = null
);
```

@correlationID: This value MUST be ignored by the server.

@partitionID: A partition identifier used to filter the current request. This value MUST NOT be NULL or empty.

@entityID: The unique identifier of the viewer entity for whom the activity events are to be retrieved. The value MUST NOT be NULL.

@minEventTime: The last update time of the activity event(s) retrieved MUST be greater than the datetime specified by @minEventTime. This value MUST NOT be NULL.

@batchSize: An integer specifying the maximum number of activity events to return in the result set. This value MUST be greater than 0 and less than or equal to 200.

@useStoredPreferences: A flag, which MUST be 1.

Return Values: An integer, which MUST be 0.

Result Sets:

This stored procedure MUST return a GetActivityEvents result set.

3.1.5.17 activityFeed_GetActivityEventsPublished

The activityFeed_GetActivityEventsPublished stored procedure is called to retrieve activity events owned by the specified publisher entity and newer than the specified datetime stored on the server. The results are filtered by the specified viewer's activity preferences and trimmed by the viewer rights.

```
PROCEDURE activityFeed_GetActivityEventsPublished (
@correlationID uniqueidentifier
,@publisherPartitionID uniqueidentifier
,@publisherEntityID bigint
,@viewerPartitionID uniqueidentifier
,@viewerEntityID bigint
,@viewerRights int = null
,@minEventTime datetime
```

```
,@batchSize int
,@useStoredPreferences bit = null
);
```

@correlationID: This value MUST be ignored by the server.

@publisherPartitionID: A partition identifier used to filter the current request. This value MUST NOT be NULL or empty.

@publisherEntityID: The unique identifier of the entity who owns the activity events to be retrieved. The value MUST NOT be NULL.

@viewerPartitionID: A GUID used to filter the current request. This value MUST be the same as the value for @publisherPartitionID.

@viewerEntityID: The unique identifier of the entity whose activity preferences are used to filter the activity events retrieved. This value MUST NOT be NULL. If the viewer entity specified by @viewerEntityID subscribes to the activity type of an activity event, then the activity event MUST be returned in the result set. If the viewer entity specified by @viewerEntityID does not subscribe to the activity type of an activity event, then the activity event MUST NOT be returned in the result set.

@viewerRights: A value indicating the rights of the specified entity to view the activity event. An activity event is returned in the result set if the value of the @ViewerRights matches the item privacy level of the activity event. If this value is NULL, the server MUST calculate the viewerRights based on the relationship between the publisher and viewer entity as described in the following table. If this value is NOT NULL, it MUST be listed in the following table:

Value	Description
1	The viewer entity has none of the rights mentioned in the following values.
2	The viewer entity is either the same as the publisher entity or a member of the publisher entity's colleagues.
4	The viewer entity is either the same as the publisher entity or a member of the publisher entity's workgroup colleagues.
8	The viewer entity is either the same as the publisher entity or the publisher entity's manager.
16	The viewer entity is the same as the publisher entity.

@minEventTime: The last update time of the activity event(s) retrieved MUST be greater than the datetime specified by @minEventTime. This value MUST NOT be NULL.

@batchSize: An integer specifying the maximum number of activity events to return in the result set. This value MUST be greater than 0 and less than or equal to 200.

@useStoredPreferences: A flag, which MUST be 1.

Return Values: An integer, which MUST be 0.

Result Sets:

This stored procedure MUST return a GetActivityEvents

3.1.5.18 activityFeed_GetActivityPreferences

The activityFeed_GetActivityPreferences stored procedure is called to retrieve the list of identifiers of those activity types that the specified user does not subscribe to activity feed events for.

```
PROCEDURE activityFeed_GetActivityPreferences (
@correlationID uniqueidentifier
,@partitionID uniqueidentifier
,@entityID bigint
);
```

@correlationID: This value MUST be ignored by the server.

@partitionID: A partition identifier used to filter the current request. This value MUST NOT be NULL or empty.

@entityID: The unique identifier of the user. This value MUST NOT be NULL.

Return Values: An integer, which MUST be 0.

Result Sets:

This stored procedure MUST return a GetActivityPreferences

If the specified user subscribes to activity feed events of all the activity types stored on the server, the result set MUST be empty.

3.1.5.19 activityFeed_GetActivityTemplates

The activityFeed_GetActivityTemplates stored procedure is called to retrieve the activity templates specified by the parameters. If @activityTypeID, and @activityTemplateID, and @IsMultivalued are all NULL, all activity templates are retrieved.

```
PROCEDURE activityFeed_GetActivityTemplates (
@correlationID uniqueidentifier
,@partitionID uniqueidentifier
,@activityTypeID bigint
,@activityTemplateID bigint = null
,@IsMultivalued bit = null
);
```

@correlationID: This value MUST be ignored by the server.

@partitionID: A partition identifier used to filter the current request. This value MUST NOT be NULL or empty.

@activityTypeID: If not NULL, the stored procedure MUST return only the activity templates (if any) whose activity type identifier equals @activityTypeID and whose other values meet the constraints specified by the other parameters. Otherwise, the activity type identifier of the activity templates stored on the server is not used in determining the activity templates to be returned.

@activityTemplateID: If not NULL, the stored procedure MUST return only the activity template (if any) whose activity template identifier equals @activityTemplateID, and then only if @activityTypeId is NULL or the activity template's activity type identifier equals @activityTypeId. Otherwise, the

activity template identifier of the activity templates stored on the server is not used in determining the activity templates to be returned.

@IsMultivalued: If @activityTemplateID is not NULL, @IsMultiValued MUST be ignored by the server. Otherwise, if @IsMultiValued is not NULL, the stored procedure MUST return only the activity templates (if any) whose IsMultiValued attribute equals @IsMultiValued and whose other values meet the constraints specified by the other parameters. If both @activityTemplateID and @IsMultivalued are NULL, the IsMultiValued attribute of the activity templates stored on the server is not used in determining the activity templates to be returned.

Return Values: An integer, which MUST be 0.

Result Sets:

This stored procedure MUST return a GetActivityTemplates result set.

3.1.5.20 activityFeed_GetActivityTypes

The activityFeed_GetActivityTypes stored procedure is called to retrieve the activity types specified by the parameters. If @applicationID, @activityTypeID, and @activityTypeName are all NULL, all activity types are retrieved.

```
PROCEDURE activityFeed_GetActivityTypes (
@correlationID uniqueidentifier
,@partitionID uniqueidentifier
,@applicationID bigint = null
,@activityTypeID bigint = null
,@activityTypeName nvarchar(256) = null
);
```

@correlationID: This value MUST be ignored by the server.

@partitionID: A partition identifier used to filter the current request. This value MUST NOT be NULL or empty.

@applicationID: If not NULL, the stored procedure MUST return only the activity types (if any) whose application identifier equals @applicationID and whose other values meet the constraints specified by the other parameters. Otherwise, the application identifier of the activity templates stored on the server is not used in determining the activity types to be returned.

@activityTypeID: If not NULL, the stored procedure MUST return only the activity type (if any) whose activity type identifier equals @activityTypeID, and then only if @applicationID is NULL or the activity type's application identifier equals @applicationID. Otherwise, the activity type identifier of the activity types stored on the server is not used in determining the activity types to be returned.

@activityTypeName: If @activityTypeID is not NULL, @activityTypeName MUST be ignored by the server. Otherwise, if @activityTypeName is not NULL, the stored procedure MUST return only the activity types (if any) whose activity type name equals @activityTypeName and whose other values meet the constraints specified by the other parameters. If both @activityTypeID and @activityTypeName are NULL, the activity type name of the activity types stored on the server is not used in determining the activity types to be returned.

Return Values: An integer, which MUST be 0.

Result Sets:

This stored procedure MUST return a GetActivityTypes result set.

3.1.5.21 activityFeed_GetLatestLastUpdateTimeConsolidated

The activityFeed_GetLatestLastUpdateTimePublished stored procedure is called to retrieve the most recent last update time of specified activity events that were subscribed to by anyone.

```
PROCEDURE activityFeed_GetLatestLastUpdateTimeConsolidated (
@correlationID uniqueidentifier
,@viewerPartitionID uniqueidentifier
,@applicationIDs nvarchar(1000)
,@viewerEntityID bigint = null
);
```

@correlationID: This value MUST be ignored by the server.

@viewerPartitionID: A partition identifier used to filter the current request. This value MUST NOT be NULL or empty.

@applicationIDs: A semicolon delimited list of unique identifiers of activity applications. The stored procedure MUST evaluate the last update time of all activity events whose activity type matches the activity types of the activity applications specified by this list of identifiers. If no activity events match the criteria, the result set MUST be empty. This value MUST NOT be NULL or empty.

@viewerEntityID: Reserved. The value MUST be NULL.

Return Values: An integer, which MUST be 0.

Result Sets:

This stored procedure MUST return a GetLatestLastActivityEventUpdateTime result set that MUST contain at most one record.

3.1.5.22 activityFeed_GetLatestLastUpdateTimePublished

The activityFeed_GetLatestLastUpdateTimePublished stored procedure is called to retrieve the most recent last update time of specified activity events that were published by anyone.

```
PROCEDURE activityFeed_GetLatestLastUpdateTimePublished (
@correlationID uniqueidentifier
,@publisherPartitionID uniqueidentifier
,@applicationIDs nvarchar(1000)
,@publisherEntityID bigint = null
);
```

@correlationID: This value MUST be ignored by the server.

@publisherPartitionID: A partition identifier used to filter the current request. This value MUST NOT be NULL or empty.

@applicationIDs: A list of semicolon delimited unique identifiers of activity applications. The stored procedure MUST evaluate the last update time of all activity events whose activity type matches the activity types of the activity applications specified by this list of identifiers. If no activity events match the criteria, the result set MUST be empty. This value MUST NOT be NULL or empty.

@publisherEntityID: Reserved. The value MUST be NULL.

Return Values: An integer, which MUST be 0.

Result Sets:

This stored procedure MUST return a GetLatestLastActivityEventUpdateTime result set that MUST contain at most one record.

3.1.5.23 activityFeed_InsertActivityApplications

The activityFeed_InsertActivityApplications stored procedure is called to create a new activity application on the server. The server MUST generate a new unique identifier for the activity application.

```
PROCEDURE activityFeed_InsertActivityApplications (
@correlationID uniqueidentifier
,@partitionID uniqueidentifier
,@applicationName nvarchar(256)
,@description nvarchar(1000) = null
,@homePageURL nvarchar(1000) = null
,@creationTime datetime = null
,@lastUpdateTime datetime = null
,@createdBy nvarchar(256) = null
);
```

@correlationID: This value MUST be ignored by the server.

@partitionID: A partition identifier used to filter the current request. This value MUST NOT be NULL or empty.

@applicationName: The name of the activity application. This value MUST NOT be NULL or empty. This value MUST NOT be the same as the name of any existing activity application on the server.

@description: The description of the activity application. This value MUST NOT be NULL.

@homePageURL: Reserved. This value MUST be NULL.

@creationTime: The creation time of the activity application. If @creationTime is NULL, the stored procedure MUST store the current UTC time as the creation time of the activity application on the server.

@lastUpdateTime: The last update time of the activity application. If @lastUpdateTime is NULL, the stored procedure MUST store the creation time as the last updated time of the activity application on the server.

@createdBy: The creator of the activity application.

Return Values: An integer, which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.1.5.24 activityFeed_InsertActivityEventsConsolidated

The activityFeed_InsertActivityEventsConsolidated stored procedure is called to create a new activity event that the specified entity monitors on the server. The server MUST generate a new unique identifier for the activity event.

```
PROCEDURE activityFeed_InsertActivityEventsConsolidated (
@correlationID uniqueidentifier
,@partitionID uniqueidentifier
,@entityTypeID nvarchar(50)
,@entityID bigint
,@activityTypeID bigint
,@templateVariable nvarchar(4000) = null
,@isRolledUp bit = null
,@creationTime datetime = null
,@lastUpdateTime datetime = null
);
```

@correlationID: This value MUST be ignored by the server.

@partitionID: A partition identifier used to filter the current request. This value MUST NOT be NULL or empty.

@entityTypeID: A string for the entity type. This value MUST be set to the string 'user'.

@entityID: The unique identifier of the entity that monitors the activity event. This value MUST NOT be NULL.

@activityTypeID: A unique identifier specifying the activity type of the activity event. This value MUST NOT be NULL.

@templateVariable: A string specifying the details of the activity event, which MUST conform to the ActivityTemplateVariable schema defined in section 2.2.6.4.1.

@isRolledUp: A flag indicating whether the activity event is a roll-up of multiple activity events of the specified activity type. If the activity event being created is a roll-up of multiple activity events, this value MUST be 1. Otherwise, the value MUST be 0.

@creationTime: The creation time of the activity event. If @creationTime is NULL, the stored procedure MUST store the current UTC time as the creation time for the activity event on the server.

@lastUpdateTime: The time of occurrence of the change event (2). If @lastUpdateTime is NULL, the stored procedure MUST store the @creationTime as the last update time for the activity event on the server.

Return Values: An integer, which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.1.5.25 activityFeed_InsertActivityEventsPublished

The activityFeed_InsertActivityEventsPublished stored procedure is called to create a new activity event that the specified entity owns on the server. The server MUST generate a new unique identifier for the activity event.

```
PROCEDURE activityFeed_InsertActivityEventsPublished (
@correlationID uniqueidentifier
,@partitionID uniqueidentifier
,@entityTypeID nvarchar(50)
,@entityID bigint
,@activityTypeID bigint
,@itemPrivacy int
```

```
,@templateVariable nvarchar(4000) = null
,@isRolledUp bit = null
,@creationTime datetime = null
,@lastUpdateTime datetime = null
);
```

@correlationID: This value MUST be ignored by the server.

@partitionID: A partition identifier used to filter the current request. This value MUST NOT be NULL or empty.

@entityTypeID: A string for the entity type. This value MUST be set to the string 'user'.

@entityID: The unique identifier of the entity that owns the activity event. This value MUST NOT be NULL.

@activityTypeID: A unique identifier specifying the activity type of the activity event. This value MUST NOT be NULL.

@itemPrivacy: An integer, specifying the set of users that have access to the activity event, which MUST conform to the Privacy type defined in section 2.2.1.5.

@templateVariable: A string specifying the details of the activity event, which MUST conform to the ActivityTemplateVariable schema defined in section 2.2.6.4.1.

@isRolledUp: A flag indicating whether the activity event is a roll-up of multiple activity events of the specified activity type. If the activity event being created is a roll-up of multiple activity events, this value MUST be 1. Otherwise, the value MUST be 0.

@creationTime: The creation time of the activity event. If @creationTime is NULL, the stored procedure MUST store the current UTC time as the creation time for the activity event on the server.

@lastUpdateTime: The time of occurrence of the change event (2). If @lastUpdateTime is NULL, the stored procedure MUST store the @creationTime as the last update time for the activity event on the server.

Return Values: An integer which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.1.5.26 activityFeed_InsertActivityTemplates

The activityFeed_InsertActivityTemplates stored procedure is called to create a new activity template on the server. The server MUST generate a new unique identifier for the activity template.

```
PROCEDURE activityFeed_InsertActivityTemplates (
@correlationID uniqueidentifier
,@partitionID uniqueidentifier
,@activityTypeID bigint
,@titleFormatLocStringName nvarchar(256)
,@titleFormatLocStringResourceFile nvarchar(256)
,@dataFormatLocStringId int = null
,@IsMultivalued bit
,@creationTime datetime = null
,@lastUpdateTime datetime = null
,@createdBy nvarchar(256) = null
```

```
);
```

@correlationID: This value MUST be ignored by the server.

@partitionID: A partition identifier used to filter the current request. This value MUST NOT be NULL or empty.

@activityTypeID: A unique identifier specifying the activity type of the activity template. This value MUST NOT be NULL.

@titleFormatLocStringName: The key string for looking up (in the title format string resource file) the localized template string to be used when generating a human-readable description of an event of the associated activity type. This value MUST NOT be NULL or empty.

@titleFormatLocStringResourceFile: The name of the resource file containing the localized template string to be used when generating a human-readable description of an event of the associated activity type. This value MUST NOT be NULL or empty.

@dataFormatLocStringId: Reserved. This value MUST be NULL.

@IsMultivalued: The IsMultivalued attribute of the activity template. This value MUST be 0 if the template takes a single value; otherwise (that is, it takes a list of values) it MUST be 1.

@creationTime: The creation time of the activity template. If @creationTime is NULL, the stored procedure MUST store the current UTC time as the creation time of the activity template on the server.

@lastUpdateTime: The last update time of the activity template. If @lastUpdateTime is NULL, the stored procedure MUST store the creation time as the last updated time of the activity template on the server.

@createdBy: This value specifies the createdBy and modifiedBy attributes of the activity template.

Return Values: An integer which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.1.5.27 activityFeed_InsertActivityTypes

The activityFeed_InsertActivityTypes stored procedure is called to create a new activity type on the server. The server MUST generate a new unique identifier for the activity type.

```
PROCEDURE activityFeed_InsertActivityTypes (
@correlationID uniqueidentifier
,@partitionID uniqueidentifier
,@applicationID bigint
,@activityTypeName nvarchar(256)
,@activityTypeNameLocStringResourceFile nvarchar(256)
,@activityTypeNameLocStringName nvarchar(256)
,@description nvarchar(1000) = null
,@logoURL nvarchar(1000) = null
,@allowRollup bit = null
,@isConsolidated bit = null
,@isPublished bit = null
,@gatheringInterval int = null
,@itemTimeToLive int = null
```

```
,@creationTime datetime = null
,@lastUpdateTime datetime = null
,@createdBy nvarchar(256) = null
);
```

@correlationID: This value MUST be ignored by the server.

@partitionID: A partition identifier used to filter the current request. This value MUST NOT be NULL or empty.

@applicationID: The application identifier of the activity type. This value MUST NOT be NULL.

@activityTypeName: The name of the activity type. This value MUST NOT be NULL or empty. The @activityTypeName also MUST NOT equal the activity type name of any existing activity type on the server.

@activityTypeNameLocStringResourceFile: The name of the resource file containing the localized activity type name. This value MUST NOT be NULL or empty.

@activityTypeNameLocStringName: The key string for looking up the localized activity type name in the activity type name localized string resource file. This value MUST NOT be NULL or empty.

@description: The description of the activity type.

@logoURL: Reserved. This value MUST be NULL.

@allowRollup: This value specifies the allowRollup attribute (sections 2.2.4.14 and 3.1.1) for the new activity type. This value MUST be 1 if the roll-ups are allowed; otherwise it MUST be 0. A NULL value is equivalent to 0.

@isConsolidated: A flag indicating whether activity events of this type are to be pre-consolidated for subscribing entities, so that they can be immediately consumed by applications. If so, this value MUST be 1; otherwise it MUST be 0. A NULL value is equivalent to 0.

@isPublished: Reserved. This value MUST be 1.

@gatheringInterval: Reserved. This value MUST be NULL.

@itemTimeToLive: Reserved. This value MUST be NULL.

@creationTime: The creation time of the activity type. If @creationTime is NULL, the stored procedure MUST store the current UTC time as the creation time of the activity type on the server.

@lastUpdateTime: The last update time of the activity type. If @lastUpdateTime is NULL, the stored procedure MUST store the creation time as the last updated time of the activity type on the server.

@createdBy: This value specifies the createdBy and modifiedBy attributes of the activity type.

Return Values: An integer, which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.1.5.28 activityFeed_InsertIfNotExistsActivityPreference

The activityFeed_InsertIfNotExistsActivityPreference stored procedure is called to create a new exclusion of the specified activity type from the activity feed subscription of the specified user. If the exclusion of the specified activity type from the activity feed subscription of the specified user already exists on the server, the state of the server MUST NOT change.

```
PROCEDURE activityFeed_InsertIfNotExistsActivityPreference (
@correlationID uniqueidentifier
,@partitionID uniqueidentifier
,@entityID bigint
,@activityTypeID bigint
);
```

@correlationID: This value MUST be ignored by the server.

@partitionID: A partition identifier used to filter the current request. This value MUST NOT be NULL or empty.

@entityID: The unique identifier of the user who does not subscribe to activity feed events for the specified activity type. This value MUST NOT be NULL.

@activityTypeID: The unique identifier of the activity type that the specified user is unsubscribing to activity feed events for. This value MUST NOT be NULL.

Return Values: An integer, which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.1.5.29 activityFeed_UpdateActivityApplications

The activityFeed_UpdateActivityApplications stored procedure is called to update the activity application specified by the unique identifier contained in @applicationID. If the activity application specified by @applicationID does not exist on the server, then the stored procedure MUST NOT make any changes to the server.

```
PROCEDURE activityFeed_UpdateActivityApplications (
@correlationID uniqueidentifier
,@partitionID uniqueidentifier
,@applicationID bigint
,@lastUpdateTime datetime
,@applicationName nvarchar(256) = null
,@description nvarchar(1000) = null
,@homePageURL nvarchar(1000) = null
,@isSoftDeleted bit = null
,@modifiedBy nvarchar(256) = null
);
```

@correlationID: This value MUST be ignored by the server.

@partitionID: A partition identifier used to filter the current request. This value MUST NOT be NULL or empty.

@applicationID: The unique identifier of the activity application to be updated. This value MUST not be NULL.

@lastUpdateTime: The last update time of the activity application. This value MUST be the current UTC time.

@applicationName: The updated name of the activity application. If @applicationName is NULL, the name of the activity application stored on the server MUST NOT be updated.

@description: The updated description of the activity application. If @description is NULL, the description of the activity application stored on the server MUST NOT be updated.

@homePageURL: Reserved. This value MUST be NULL.

@isSoftDeleted: Reserved. This value MUST be NULL.

@modifiedBy: The last modifier of the activity application. If @modifiedBy is NULL, the modifier of the activity application stored on the server MUST NOT be updated.

Return Values: An integer, which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.1.5.30 activityFeed_UpdateActivityEventsConsolidated

The activityFeed_UpdateActivityEventsConsolidated stored procedure is called to update the specified activity event on the server. If the activity event does not exist on the server, the state of the server MUST not change.

```
PROCEDURE activityFeed_UpdateActivityEventsConsolidated (
@correlationID uniqueidentifier
,@partitionID uniqueidentifier
,@activityEventID bigint
,@lastUpdateTime datetime
,@templateVariable nvarchar(4000)
,@isRolledUp bit
);
```

@correlationID: This value MUST be ignored by the server.

@partitionID: A partition identifier used to filter the current request. This value MUST NOT be NULL or empty.

@activityEventID: The unique identifier of the activity event that is to be updated. This value MUST NOT be NULL.

@lastUpdateTime: The time of occurrence of the change event (2). The value MUST NOT be NULL and MUST be greater than the value of LastUpdateTime stored on the server for the activity event.

@templateVariable: A string specifying the details of the activity event which MUST conform to the ActivityTemplateVariable schema defined in section 2.2.6.4.1.

@isRolledUp: A flag indicating whether the activity event is a roll-up of multiple activity events of the specified activity type. If the activity event being created is a roll-up of multiple activity events, this value MUST be 1. Otherwise, the value MUST be 0.

Return Values: An integer, which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.1.5.31 activityFeed_UpdateActivityEventsPublished

The activityFeed_UpdateActivityEventsPublished stored procedure is called to update the specified activity event on the server. If the activity event does not exist on the server, the state of the server MUST not change.

```
PROCEDURE activityFeed_UpdateActivityEventsPublished (
@correlationID uniqueidentifier
,@partitionID uniqueidentifier
,@activityEventID bigint
,@templateVariable nvarchar(4000)
,@lastUpdateTime datetime
,@itemPrivacy int
,@isRolledUp bit
);
```

@correlationID: This value MUST be ignored by the server.

@partitionID: A partition identifier used to filter the current request. This value MUST NOT be NULL or empty.

@activityEventID: The unique identifier of the activity event that is to be updated. This value MUST NOT be NULL.

@templateVariable: A string specifying the details of the activity event, which MUST conform to the ActivityTemplateVariable schema defined in section 2.2.6.4.1.

@lastUpdateTime: The time of occurrence of the change event (2). The value MUST NOT be NULL and MUST be greater than the value of LastUpdateTime stored on the server for the activity event.

@itemPrivacy: An integer, specifying the set of users that have access to the activity event, which MUST conform to the Privacy type defined in section 2.2.1.5.

@isRolledUp: A flag indicating whether the activity event is a roll-up of multiple activity events of the specified activity type. If the activity event being created is a roll-up of multiple activity events, this value MUST be 1. Otherwise, the value MUST be 0.

Return Values: An integer, which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.1.5.32 activityFeed_UpdateActivityTemplates

The activityFeed_UpdateActivityTemplates stored procedure is called to update the activity template specified by the unique identifier contained in @activityTemplateID on the server. If the activity template specified by @activityTemplateID does not exist on the server, then the stored procedure MUST NOT make any changes to the server.

```
PROCEDURE activityFeed_UpdateActivityTemplates (
@correlationID uniqueidentifier
,@partitionID uniqueidentifier
,@activityTemplateID bigint
,@lastUpdateTime datetime
,@titleFormatLocStringName nvarchar(256) = null
,@titleFormatLocStringResourceFile nvarchar(256) = null
,@dataFormatLocStringId int = null
```

```
,@IsMultivalued bit = null
,@modifiedBy nvarchar(256) = null
);
```

@correlationID: This value MUST be ignored by the server.

@partitionID: A partition identifier used to filter the current request. This value MUST NOT be NULL or empty.

@activityTemplateID: The unique identifier of the activity template to be updated. This value MUST not be NULL.

@lastUpdateTime: The last update time of the activity template. This value MUST be the current UTC time.

@titleFormatLocStringName: The updated key string for looking up (in the title format string resource file) the localized template string to be used when generating a human-readable description of an event of the associated activity type. If @titleFormatLocStringName is NULL, the title format localized string name of the activity template stored on the server MUST NOT be updated.

@titleFormatLocStringResourceFile: The updated name of the resource file containing the localized template string to be used when generating a human-readable description of an event of the associated activity type. @titleFormatLocStringResourceFile is NULL, the title format localized string resource file name of the activity template stored on the server MUST NOT be updated.

@dataFormatLocStringId: Reserved. This value MUST be NULL.

@IsMultivalued: The IsMultivalued attribute of the activity template. This value MUST be 0 if the template takes a single value. If the template takes a list of values, it MUST be 1. If @IsMultivalued is NULL, the IsMultivalued attribute of the activity template stored on the server MUST NOT be updated.

@modifiedBy: The login of the user updating the activity template. If @modifiedBy is NULL, the last modifier of the activity template stored on the server MUST NOT be updated.

Return Values: An integer, which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.1.5.33 activityFeed_UpdateActivityTypes

The activityFeed_UpdateActivityTypes stored procedure is called to update the activity type specified by the unique identifier contained in @activityTypeID on the server. If the activity type specified by @activityTypeID does not exist on the server, then the stored procedure MUST NOT make any changes to the server.

```
PROCEDURE activityFeed_UpdateActivityTypes (
@correlationID uniqueidentifier
,@partitionID uniqueidentifier
,@activityTypeID bigint
,@activityTypeName nvarchar(256) = null
,@activityTypeNameLocStringResourceFile nvarchar(256) = null
,@activityTypeNameLocStringName nvarchar(256) = null
,@description nvarchar(1000) = null
,@logoURL nvarchar(1000) = null
```

```
,@allowRollup bit = null
,@isConsolidated bit = null
,@isPublished bit = null
,@gatheringInterval int = null
,@itemTimeToLive int = null
,@modifiedBy nvarchar(256) = null
);
```

@correlationID: This value MUST be ignored by the server.

@partitionID: A partition identifier used to filter the current request. This value MUST NOT be NULL or empty.

@activityTypeID: The unique identifier of the activity type to be updated. This value MUST not be NULL.

@activityTypeName: The updated name for the activity type. If this value is NULL, the activity type name MUST NOT be changed on the server.

@activityTypeNameLocStringResourceFile: The updated name of the resource file containing the localized activity type name. If this value is NULL, the activity type name localized string resource file name MUST NOT be changed on the server.

@activityTypeNameLocStringName: The updated key string for looking up the localized activity type name in the activity type name localized string resource file. If this value is NULL, the activity type name localized string name MUST not be changed on the server.

@description: The updated description for the activity type. If this value is NULL, the activity type's value MUST NOT be changed on the server.

@logoURL: Reserved. This value MUST be NULL.

@allowRollup: The updated allowRollup attribute (sections 2.2.4.14 and 3.1.1) for the activity type. This value MUST be 1 if the roll-ups are allowed; otherwise it MUST be 0. If this value is NULL, the activity type's value MUST NOT be changed on the server.

@isConsolidated: A updated flag indicating whether activity events of this type are to be pre-consolidated for subscribing entities, so that they can be immediately consumed by applications. If so, this value MUST be 1; otherwise it MUST be 0. If this value is NULL, the isConsolidated attribute MUST NOT be changed on the server.

@isPublished: Reserved. This value MUST be NULL.

@gatheringInterval: Reserved. This value MUST be NULL.

@itemTimeToLive: Reserved. This value MUST be NULL.

@modifiedBy: The login of the user updating the activity type. If @modifiedBy is NULL, the last modifier of the activity template stored on the server MUST NOT be updated.

Return Values: An integer, which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.1.5.34 profile_GetUsersColleaguesAndRights

The profile_GetUsersColleaguesAndRights stored procedure is called to retrieve information about specified users, those users' colleagues, and the visibility of those colleagues' events to the associated user.

```
PROCEDURE profile_GetUsersColleaguesAndRights (
@partitionID uniqueidentifier
,@KeyValue nvarchar(4000)
,@correlationId uniqueidentifier = null
);
```

@partitionID: A partition identifier used to filter the current request. This value MUST NOT be NULL or empty.

@KeyValue: A semicolon delimited list of unique identifiers of user profiles whose information is to be retrieved. This value MUST NOT be NULL or empty.

@correlationId: This value MUST be ignored by the server.

Return Values: An integer, which MUST be 0.

Result Sets:

This stored procedure MUST return a GetUserInfo result set.

This stored procedure MUST return a GetUsersColleaguesAndRights result set.

3.1.5.35 profile_GetViewerPublisherInfo

The profile_GetViewerPublisherInfo stored procedure is called to retrieve information about the specified viewer and publisher users.

```
PROCEDURE profile_GetViewerPublisherInfo (
@partitionID uniqueidentifier
,@viewer nvarchar(400) = null
,@publisher nvarchar(400) = null
,@correlationId uniqueidentifier = null
);
```

@partitionID: A GUID used to filter the current request. This value MUST NOT be NULL or empty.

@viewer: The user name of the viewer's user profile. If this value is NULL, the viewer result set will be empty.

@publisher: The user name of the publisher's user profile. If this value is NULL, the publisher result set will be empty.

@correlationId: This value MUST be ignored by the server.

Return Values: An integer, which MUST be 0.

Result Sets:

This stored procedure MUST return one GetUserInfo result set for the viewer followed by one GetUserInfo result set for the publisher.

3.1.5.36 profile_GetUserEventsIDRange

The profile_GetUserEventsIDRange stored procedure is called to retrieve the unique identifiers of the oldest and latest activity events stored on the server that were last updated after the specified @previousRunTime.

```
PROCEDURE profile_GetUserEventsIDRange (
@partitionID uniqueidentifier
,@previousRunTime datetime
,@currentRunTime datetime
,@correlationId uniqueidentifier = null
);
```

@partitionID: A partition identifier used to filter the current request. This value MUST NOT be NULL or empty.

@previousRunTime: The datetime specifying that the last update time of the oldest activity event MUST be greater than @previousRunTime. This value MUST NOT be NULL.

@currentRunTime: Reserved. This value MUST be ignored by the server.

@correlationId: This value MUST be ignored by the server.

Return Values: An integer, which MUST be 0.

Result Sets:

This stored procedure MUST return a GetUserEventsIDOldest result set, which MUST contain at most one record.

This stored procedure MUST return a GetUserEventsIDNewest result set, which MUST contain at most one record.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

3.2 Client Details

None.

3.2.1 Abstract Data Model

None.

3.2.2 Timers

None.

3.2.3 Initialization

None.

3.2.4 Higher-Layer Triggered Events

None.

3.2.5 Message Processing Events and Sequencing Rules

None.

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

4 Protocol Examples

In this example, the user profile store contains five user profiles representing five different users.

User	Colleague List
1	2, 3
2	1, 4, 5
3	1
4	2, 5
5	2, 4

The property store contains four properties and their Privacy Policy Type values.

Property Name	Type	State
Name	String	Enabled
Address	String	Enabled
Birthday	Date	Enabled
Marriage Date	Date	Enabled
Interests	GUID	Enabled
Experts	GUID	Enabled

The user profile property store contains nine user profile property values.

User	Property	Value	Privacy
1	Name	User1	Public
1	Address	123 New Road, New City, ST	Manager
1	Marriage Data	02/29/2008	Public
1	Interests	Movies	Public
1	Experts	Movies	Public
2	Name	One More User2	Public
3	Name	Different User3	Public
3	Birthday	01/02/1973	Contacts
4	Name	Another User4	Public
4	Marriage Data	02/03/1974	Private
5	Name	Last User5	Public

User	Property	Value	Privacy
5	Address	456 Some Road, Some City, ST	Organization
5	Interests	Movies	Public

The user profile change entry log contains five user profile change entries.

EventId	User	Change Type	Object Type	Property Name	Value	Change Time
1	1	Modify	SingleValue Property	Address	123 New Road, New City, ST	02/13/2008 1:23:45 PM
2	2	Add	Colleague		Another User4	02/13/2008 2:34:56 PM
3	4	Add	Colleague		One More User2	02/13/2008 2:34:56 PM
4	5	Add	WebLog		<?xml version="1.0" encoding="utf-16"?> <WebLog> <Title> My New Post </Title> <Permalink> http://site/p5/newpost </Permalink> </WebLog>	02/13/2008 3:45:07 PM
5	3	Remove	WebLog		<?xml version="1.0" encoding="utf-16"?> <WebLog> <Title> My Old Post </Title> <Permalink> http://site/p3/oldpost </Permalink> </WebLog>	02/13/2008 4:56:18 PM
6	1	Add	SingleValue Property	Marriage Data	02/29/2008	03/01/2008 3:21:17 PM

4.1 Retrieve All Changes for the Colleagues of a User

One of the possible scenarios is when it is necessary to retrieve user profile change events of the Colleague properties of a specified user. The following steps can be taken.

- With the user name use the **membership_getColleagueSuggestions** stored procedure (specified in [MS-UPSPROF2], section 3.1.5.3) to get the user identifier (**RecordId**).
- Call `profile_GetUserColleagueEvents (partitionID, RecordId, Date, ChangeTypeMask, ObjectTypeMask, correlationId )`

For example, suppose an operation to get all "Add" and "Modify" events for the Colleague properties of "User1" since "02/13/2008 2:00:00 PM". The steps to be taken would be:

- Get "1" from the user profile store using membership_getColleagueSuggestions [MS-UPSPROF2].
- Call `profile_GetUserColleagueEvents ('0c37852b-34d0-418e-91c6-2ac25af4be5b', 1, '02/13/2008 2:00:00 PM', 0x03, 0x03FF, NULL).`

In this example, a simplified version of the result set is returned from **profile_GetUserColleagueEvents**, with only a subset of the columns

The returned result set would consist of (Non-relevant columns have been omitted for clarity.):

EventId	User	Change Type	ObjectType	Value	Change Time
2	2	Add	Colleague	Another User4	02/13/2008 2:34:56 PM

4.2 Retrieve All Changes for a Specified User

Using the stored procedures in this protocol, it is possible to retrieve the user profile change events that have occurred for a specific **user**.

This can be accomplished by calling **profile_GetUserEvents**. For example:

- Retrieve the RecordId for user 1.
- Retrieve the ViewerRights for user 1.
- Call GetUserEvents with these parameters.

Parameter	Value
@partitionID	`0c37852b-34d0-418e-91c6-2ac25af4be5b`
@RecordId	The RecordId for user 1
@ViewerRights	The ViewerRights for user 1
@MinEventId	NULL
@MinEventTime	02/01/2008 12:00:00 AM
@ChangeTypeMask	0x07
@ObjectTypeMask	0x0100
@sortDescending	0
@correlationId	NULL

- This call would return the following result set. (Non-relevant columns have been omitted for clarity.)

RecordId	Change Type	EventTime	NewValueData	ItemSecurity
RecordId for user 1	0x02	02/13/2008 1:23:45 PM	123 New Road, New City, ST	0x08
RecordId for user 1	0x01	03/01/2008 3:21:17 PM	02/29/2008	0x01

4.3 Synchronization

This protocol can be used to maintain the data synchronization between two or more servers. By calling some of these protocol procedures periodically, it is possible for a server to keep track of the latest changes on the other servers. In this example, Server A gets the most recent changes from Server B. The following steps are applied:

- Server A retrieves the date and time of its most recent user profile change event (**lastChangeTime**).
- Call `profile_GetUserEvents (0c37852b-34d0-418e-91c6-2ac25af4be5b, NULL, NULL, NULL, lastChangeTime, 0x0F, 0x03FF, NULL)`.
- If the number of user profile change events returned is less than 1000, then Server A applies those records. Otherwise, Server A synchronizes all of the data with Server B.

Suppose that Server B contains all the data from the example data section, and Server A contains only the first three user profile change events in its user profile change log. The steps to be taken to synchronize Server A and Server B would be as follows:

- Server A gets "02/13/2008 2:23:56 PM" from **user profile change event** with the **EventId** 3.
- Server A calls `profile_GetUserEvents (0c37852b-34d0-418e-91c6-2ac25af4be5b, NULL, NULL, NULL, "02/13/2008 2:23:56 PM", 0x0F, 0x03FF, NULL)` on Server B.

The following result set would be returned.

In this example, only a subset of the returned columns is being displayed.

EventId	User	Change Type	ObjectType	Value	Change Time
4	5	Add	WebLog	`<?xml version="1.0" encoding="utf-16"?> <WebLog> <Title> My New Post </Title> <Permalink> http://site/p5/newpost </Permalink> </WebLog>`	02/13/2008 3:45:07 PM
5	3	Remove	WebLog	`<?xml version="1.0" encoding="utf-16"?> <WebLog> <Title> My Old Post`	02/13/2008 4:56:18 PM

EventId	User	Change Type	ObjectType	Value	Change Time
				</Title> <Permalink> http://site/p3/oldpost </Permalink> </WebLog>	

4.4 Recording a User Profile Change Event in the Activity Feed

Using the stored procedures in this protocol, it is possible to create a feed of events that have happened for users and their colleagues. To create such a list the following sequence of requests could be made:

- Call activityFeed_GetActivityApplications with the following parameters to find all of the activity search components.

Parameter	Value
@correlationID	00000000-0000-0000-0000-000000000000
@partitionID	0c37852b-34d0-418e-91c6-2ac25af4be5b
@applicationID	NULL
@applicationName	NULL

- This call would return the following result set (non-relevant columns have been omitted for clarity).

ApplicationId
1
2
3

- Knowing the ApplicationId values for the search component applications, call activityFeed_GetLatestLastUpdateTimeConsolidated with the following parameters to find out the last time the consolidated events were updated.

Parameter	Value
@viewerPartitionId	0c37852b-34d0-418e-91c6-2ac25af4be5b
@applicationIDs	1;2;3
@CorrelationId	00000000-0000-0000-0000-000000000000

- This call would return the following results.

LastUpdateTime
2008-02-12 23:48:30.927

- Knowing the ApplicationId values for the search component applications, call activityFeed_GetLatestLastUpdateTimePublished with the following parameters to find out the last time the published events were updated.

Parameter	Value
@publisherPartitionId	0c37852b-34d0-418e-91c6-2ac25af4be5b
@applicationIDs	1;2;3
@CorrelationId	00000000-0000-0000-0000-000000000000

- This call will return the following result set.

LastUpdateTime
2008-02-12 23:48:30.927

- To publish an event for user 1, make a call to dbo.activityFeed_InsertActivityEventsPublished with the following values.

Parameter	Value
@partitionId	0c37852b-34d0-418e-91c6-2ac25af4be5b
@EntityTypeId	user
@EntityId	1
@ActivityTypeId	1
@CreationTime	2010-01-06 23:51:02.1600000
@LastUpdateTime	2008-02-13 15:45:07.9370000
@TemplateVariable	<?xml version="1.0"?> <ActivityTemplateVariable xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns="AF"> <Owner xmlns=""> <AccountName>User1</AccountName> <Email>User1</Email> <Type>user</Type> <Name>User1</Name> <Href>http://server/my/person.aspx?accountname=User1</Href> <Id>1</Id> </Owner> <Publisher xmlns=""> <AccountName>User1</AccountName> <Email>User1</Email> <Type>user</Type> <Name>User1</Name> <Href>http://server/my/person.aspx?accountname=User1</Href> <Id>1</Id>

Parameter	Value
	</Publisher> <PublishDate xmlns="">2010-01-06T23:47:42.937</PublishDate> <Name xmlns="">Address</Name> <Value xmlns="">123 New Road, New City, ST</Value> </ActivityTemplateVariable>
@IsRolledUp	0
@ItemPrivacy	2
@CorrelationId	00000000-0000-0000-0000-000000000000

- The call would not return a result set.
- To insert the same event into the consolidated feed of user 2, make a call to activityFeed_InsertActivityEventsConsolidated with the following parameters.

Parameter	Value
@partitionId	0c37852b-34d0-418e-91c6-2ac25af4be5b
@EntityTypeId	User
@EntityId	2
@ActivityTypeId	1
@CreationTime	2010-01-06 23:51:02.1600000
@LastUpdateTime	2008-02-13 15:45:07.9370000
@TemplateVariable	<?xml version="1.0"?> <ActivityTemplateVariable xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns="AF"> <Owner xmlns=""> <AccountName>User2</AccountName> <Email>User2</Email> <Type>user</Type> <Name>User2</Name> <Href>http://server/my/person.aspx?accountname=User2</Href> <Id>2</Id> </Owner> <Publisher xmlns=""> <<AccountName>User1</AccountName> <Email>User1</Email> <Type>user</Type> <Name>User1</Name> <Href>http://server/my/person.aspx?accountname=User1</Href> <Id>1</Id> </Publisher> <PublishDate xmlns="">2010-01-06T23:47:42.953</PublishDate> <Name xmlns="">Address</Name>

Parameter	Value
	<Value xmlns="">123 New Road, New City, ST</Value> </ActivityTemplateVariable>
@IsRolledUp	0
@ItemPrivacy	2
@CorrelationId	00000000-0000-0000-0000-000000000000

- This call would not return a result set.

4.5 Retrieving Information from a User's Published Feed

Following the previous example, this example will go through how the activity feed sprocs can be used to retrieve data.

- Make a call to activityFeed_GetActivityEventsPublished with the following parameters.

Parameter	Value
@viewerPartitionId	0c37852b-34d0-418e-91c6-2ac25af4be5b
@viewerEntityId	2
@publisherPartitionId	0c37852b-34d0-418e-91c6-2ac25af4be5b
@publisherEntityId	1
@viewerRights	1
@useStoredPreferences	1
@minEventTime	2005-12-23 23:53:24.1500000
@batchSize	10
@CorrelationId	F4374223-B76D-4971-95E5-3C0714A8F864

- The call will return the following data set. For clarity, it has been split across two tables. The first table (following) covers the identifying information which will be used later to understand the TemplateVariable, and the second table contains the TemplateVariable

PartitionId	ActivityEventId	EntityTypeId	EntityId	ActivityTypeId	CreationTime	LastUpdateTime	IsRolledUp
0c37852b-34d0-418e-91c6-2ac25af4be5b	3	user	2	1	2010-01-06 23:30:46.700	2010-01-06 23:26:40.880	0

TemplateVariable

```
<?xml version="1.0"?>
<ActivityTemplateVariable xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns="AF">
  <Owner xmlns="">
    <AccountName>User1</AccountName>
    <Email>User1</Email>
    <Type>user</Type>
    <Name>User1</Name>
    <Href>http://server/my/person.aspx?accountname=User1</Href>
    <Id>1</Id>
  </Owner>
  <Publisher xmlns="">
    <AccountName>User1</AccountName>
    <Email>User1</Email>
    <Type>user</Type>
    <Name>User1</Name>
    <Href>http://server/my/person.aspx?accountname=User1</Href>
    <Id>1</Id>
  </Publisher>
  <PublishDate xmlns="">2010-01-06T23:47:42.937</PublishDate>
  <Name xmlns="">Address</Name>
  <Value xmlns="">123 New Road, New City, ST</Value>
</ActivityTemplateVariable>
```

- Now that the data has been retrieved, to understand it there are several calls to find the template information for this event type. First, a call to activityFeed_GetActivityTypes with the following parameters.

Parameter	Value
@partitionId	0c37852b-34d0-418e-91c6-2ac25af4be5b
@CorrelationId	F4374223-B76D-4971-95E5-3C0714A8F864

- This call would return the following data set (some values have been omitted for clarity).

ActivityTypeId	ApplicationId	ActivityTypeName	Activity Type Name Loc String Resource File	ActivityTypeNameLocStringName
15	2	SocialRatings	OsrvCore	ActivityFeed_SocialRatings_Type_Display
16	3	INTERNAL_ChangeMarker	OsrvCore	ActivityFeed_INTERNAL_ChangeMarker_Type_Display
17	1	StatusMessage	OsrvCore	ActivityFeed_Status_Message_Type_Display
4	1	WorkplaceAnniversary_	OsrvCo	ActivityFeed_WorkplaceAnniversary_Remin

ActivityTypeId	ApplicationId	ActivityTypeName	Activity Type Name Loc String Resource File	ActivityTypeNameLocStringName
		Reminder	re	der_Type_Display
5	1	WorkplaceAnniversary_Today	OsrvCore	ActivityFeed_WorkplaceAnniversary_Today_Type_Display
6	1	ColleagueAddition	OsrvCore	ActivityFeed_ColleagueAddition_Type_Display
7	1	TitleChange	OsrvCore	ActivityFeed_TitleChange_Type_Display
8	1	ManagerChange	OsrvCore	ActivityFeed_ManagerChange_Type_Display
9	1	BlogUpdate	OsrvCore	ActivityFeed_BlogUpdate_Type_Display
10	1	DLMembershipChange	OsrvCore	ActivityFeed_DLMembershipChange_Type_Display
11	1	SharingInterest	OsrvCore	ActivityFeed_SharingInterest_Type_Display
12	2	SocialTaggingByColleague	OsrvCore	ActivityFeed_SocialTaggingByColleague_Type_Display
13	2	NoteboardPosts	OsrvCore	ActivityFeed_NoteboardPosts_Type_Display
14	2	SocialTaggingByAnyone	OsrvCore	ActivityFeed_SocialTaggingByAnyone_Type_Display
1	1	ProfilePropertyChange	OsrvCore	ActivityFeed_ProfilePropertyChange_Type_Display
2	1	Birthday_Reminder	OsrvCore	ActivityFeed_Birthday_Reminder_Type_Display
3	1	Birthday_Today	OsrvCore	ActivityFeed_Birthday_Today_Type_Display

- To get the actual Template, make a call to activityFeed_GetActivityTemplates with the following parameters.

Parameter	Value
@partitionId	0c37852b-34d0-418e-91c6-2ac25af4be5b
@ActivityTypeId	1
@CorrelationId	F4374223-B76D-4971-95E5-3C0714A8F864

- This call would return the following data set (some columns omitted for clarity).

Activity TemplateId	Activity TypeId	TitleFormatLoc StringName	TitleFormat LocString ResourceFile
2	1	ActivityFeed_ProfilePropertyChange_MV_Template	OsrvCore
1	1	ActivityFeed_ProfilePropertyChange_SV_Template	OsrvCore

- From this, there is enough information to take the Template Variable returned by activityFeed_GetActivityEventsPublished and make it into a more user friendly format.

4.6 Retrieving Information from a User's Consolidated Feed

Taking into consideration the template information retrieved in the previous example, it is also possible to use the stored procedures in this protocol to get data from a user's consolidated feed, in essence showing them information about what their colleagues have been doing.

- Make a call to activityFeed_GetActivityEventsConsolidated with the following parameters.

Parameter	Value
@partitionId	0c37852b-34d0-418e-91c6-2ac25af4be5b
@EntityId	2
@useStoredPreferences	1
@minEventTime	2005-12-23 23:53:24.1500000
@batchSize	40
@CorrelationId	F4374223-B76D-4971-95E5-3C0714A8F864

- This call would return the following data set. Some columns have been omitted for clarity. Appropriate metadata about the TemplateVariable is included in the first table, and the TemplateVariable itself is contained in the second table.

PartitionId	ActivityEventId	EntityTypeId	EntityId	ActivityTypeId	CreationTime	LastUpdateTime	IsRolledUp
0c37852b-34d0-418e-91c6-2ac25af4be5b	3	user	2	1	2010-01-06 23:30:46.700	2010-01-06 23:26:40.880	0

TemplateVariable
<?xml version="1.0"?> <ActivityTemplateVariable xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns="AF">

TemplateVariable
<Owner xmlns=""> <AccountName>User2</AccountName> <Email>User2</Email> <Type>user</Type> <Name>User2</Name> <Href>http://server/my/person.aspx?accountname=User2</Href> <Id>2</Id> </Owner> <Publisher xmlns=""> <<AccountName>User1</AccountName> <Email>User1</Email> <Type>user</Type> <Name>User1</Name> <Href>http://server/my/person.aspx?accountname=User1</Href> <Id>1</Id> </Publisher> <PublishDate xmlns="">2010-01-06T23:47:42.953</PublishDate> <Name xmlns="">Address</Name> <Value xmlns="">123 New Road, New City, ST</Value> </ActivityTemplateVariable>

- Given that this is the same activity type as in the previous example, there is enough data to understand this event as well.

4.7 Delete old activities from the consolidated and published feeds

- To save space, this protocol can be called occasionally to trim old events from users' published and consolidated feeds. The following sprocs from the protocol would be used for this.
- To clear the consolidated feeds, make a call to activityFeed_DeleteActivityEventsConsolidated with the following parameters.

Parameter	Value
@partitionId	0c37852b-34d0-418e-91c6-2ac25af4be5b
@minEventTime	2009-12-23 23:53:24.1500000
@batchSize	1000
@CorrelationId	00000000-0000-0000-0000-000000000000

- This call would not return any results.
- To clear the published feeds, make a call to activityFeed_DeleteActivityEventsPublished with the following parameters.

Parameter	Value
@partitionId	0c37852b-34d0-418e-91c6-2ac25af4be5b
@minEventTime	2009-12-23 23:53:24.1500000
@batchSize	1000
@CorrelationId	00000000-0000-0000-0000-000000000000

This call would not return any results.

5 Security

5.1 Security Considerations for Implementers

This protocol supports the **Security Support Provider Interface (SSPI)** and SQL authentication with the protocol server role. These authentication methods are defined in [MS-TDS].

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® Server 2010

Exceptions, if any, are noted below. If a service pack or Quick Fix Engineering (QFE) number appears with the product version, behavior changed in that service pack or QFE. The new behavior also applies to subsequent service packs of the product unless otherwise specified. If a product edition appears with the product version, behavior is different in that product edition.

Unless otherwise specified, any statement of optional behavior in this specification that is prescribed using the terms SHOULD or SHOULD NOT implies product behavior in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term MAY implies that the product does not follow the prescription.

7 Change Tracking

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

8 Index

A

Abstract data model
  client 58
  server 29
activityFeed_DeleteActivityEventsConsolidated method 38
activityFeed_DeleteActivityEventsPublished method 39
activityFeed_DeleteActivityTemplates method 40
activityFeed_DeleteActivityTypes method 40
activityFeed_DeleteIfExistsActivityPreference method 40
activityFeed_GetActivityApplications method 41
activityFeed_GetActivityEventsConsolidated method 42
activityFeed_GetActivityEventsPublished method 42
activityFeed_GetActivityPreferences method 44
activityFeed_GetActivityTemplates method 44
activityFeed_GetActivityTypes method 45
activityFeed_GetLatestLastUpdateTimeConsolidated method 46
activityFeed_GetLatestLastUpdateTimePublished method 46
activityFeed_InsertActivityApplications method 47
activityFeed_InsertActivityEventsConsolidated method 47
activityFeed_InsertActivityEventsPublished method 48
activityFeed_InsertActivityTemplates method 49
activityFeed_InsertActivityTypes method 50
activityFeed_InsertIfNotExistsActivityPreference method 52
activityFeed_UpdateActivityApplications method 52
activityFeed_UpdateActivityEventsConsolidated method 53
activityFeed_UpdateActivityEventsPublished method 54
activityFeed_UpdateActivityTemplates method 54
activityFeed_UpdateActivityTypes method 55
ActivityTemplateVariable
  element 27
Applicability 10
Attribute groups - overview 28
Attributes - overview 28

B

Binary structures - overview 13
Bit fields - overview 13

C

Capability negotiation 10
Change tracking 75
Change Type simple type 11
Client
  abstract data model 58
  higher-layer triggered events 59
  initialization 59
  local events 59
  message processing 59
  sequencing rules 59
  timer events 59
  timers 58
Common data types
  overview 11
Complex types
  Entity 26
  Link 25
  List 26
Complex types - overview 25

D

Data model - abstract
  client 58
  server 29
Data types
  Change Type simple type 11
  common 11
  Object Type simple type 11
  Privacy Policy Type simple type 11
  Privacy Type simple type 13
  Value simple type 12
Data types - simple
  Change Type 11
  Object Type 11
  Privacy Policy Type 11
  Privacy Type 13
  Value 12
Delete old activities from the consolidated and published feeds example 71

E

Elements
  ActivityTemplateVariable 27
Elements - overview 27
Entity - complex type 26
EnumProfileChangesOrganization result set 19
EnumProfileChangesUser result set 16
Events
  local - client 59
  local - server 58
  timer - client 59
  timer - server 58
Examples
  delete old activities from the consolidated and published feeds 71
  overview 60
  recording a user profile change event in the activity feed 64
  retrieve all changes for a specified user 62
  retrieve all changes for the colleagues of a user 61
  retrieving information from a user's consolidated feed 70

retrieving information from a user's published feed 67
sample data 60
synchronization 63

F

Fields - vendor-extensible 10
Flag structures - overview 13

G

GetActivityApplications result set 20
GetActivityEvents result set 20
GetActivityPreferences result set 21
GetActivityTemplates result set 21
GetActivityTypes result set 22
GetCurrentChangeTokenOrganization result set 19
GetCurrentChangeTokenUser result set 17
GetInterestsExpertsEvents result set 17
GetLatestLastActivityEventUpdateTime result set 23
GetUserColleagueEvents result set 13
GetUserEvents result set 15
GetUserEventsIDNewest result set 24
GetUserEventsIDOldest result set 24
GetUserEventsNoRecordId result set 15
GetUserEventsRecordId result set 18
GetUserInfo result set 23
GetUsersColleaguesAndRights result set 24
Glossary 6
Groups - overview 28

H

Higher-layer triggered events
client 59
server 31

I

Implementer - security considerations 73
Index of security parameters 73
Informative references 7
Initialization
client 59
server 31
Introduction 6

L

Link - complex type 25
List - complex type 26
Local events
client 59
server 58

M

Message processing
client 59
server 31
Messages
ActivityTemplateVariable element 27
attribute groups 28
attributes 28
binary structures 13
bit fields 13
common data types 11
complex types 25
elements 27
Entity complex type 26
EnumProfileChangesOrganization result set 19
EnumProfileChangesUser result set 16
flag structures 13
GetActivityApplications result set 20
GetActivityEvents result set 20
GetActivityPreferences result set 21
GetActivityTemplates result set 21
GetActivityTypes result set 22
GetCurrentChangeTokenOrganization result set 19
GetCurrentChangeTokenUser result set 17
GetInterestsExpertsEvents result set 17
GetLatestLastActivityEventUpdateTime result set 23
GetUserColleagueEvents result set 13
GetUserEvents result set 15
GetUserEventsIDNewest result set 24
GetUserEventsIDOldest result set 24
GetUserEventsNoRecordId result set 15
GetUserEventsRecordId result set 18
GetUserInfo result set 23
GetUsersColleaguesAndRights result set 24
groups 28
Link complex type 25
List complex type 26
namespaces 25
simple types 25
table structures 25
transport 11
view structures 25
XML structures 25
Methods
activityFeed_DeleteActivityEventsConsolidated 38
activityFeed_DeleteActivityEventsPublished 39
activityFeed_DeleteActivityTemplates 40
activityFeed_DeleteActivityTypes 40
activityFeed_DeleteIfExistsActivityPreference 40
activityFeed_GetActivityApplications 41
activityFeed_GetActivityEventsConsolidated 42
activityFeed_GetActivityEventsPublished 42
activityFeed_GetActivityPreferences 44
activityFeed_GetActivityTemplates 44
activityFeed_GetActivityTypes 45
activityFeed_GetLatestLastUpdateTimeConsolidated 46
activityFeed_GetLatestLastUpdateTimePublished 46
activityFeed_InsertActivityApplications 47
activityFeed_InsertActivityEventsConsolidated 47
activityFeed_InsertActivityEventsPublished 48
activityFeed_InsertActivityTemplates 49
activityFeed_InsertActivityTypes 50

activityFeed_InsertIfNotExistsActivityPreference 52
activityFeed_UpdateActivityApplications 52
activityFeed_UpdateActivityEventsConsolidated 53
activityFeed_UpdateActivityEventsPublished 54
activityFeed_UpdateActivityTemplates 54
activityFeed_UpdateActivityTypes 55
profile_AddUserProfileEvent 35
profile_DeleteUserEvents 33
profile_EnumProfileChanges 36
profile_GenerateAnniversaryEvents 34
profile_GetCurrentChangeToken 35
profile_GetInterestsExpertsEvents 37
profile_GetUserColleagueEvents 33
profile_GetUserColleagueEventsByDate 38
profile_GetUserEvents 31
profile_GetUserEventsIDRange 58
profile_GetUsersColleaguesAndRights 57
profile_GetViewerPublisherInfo 57

N

Namespaces 25
Normative references 7

O

Object Type simple type 11
Overview (synopsis) 8

P

Parameters - security index 73
Preconditions 10
Prerequisites 10
Privacy Policy Type simple type 11
Privacy Type simple type 13
Product behavior 74
profile_AddUserProfileEvent method 35
profile_DeleteUserEvents method 33
profile_EnumProfileChanges method 36
profile_GenerateAnniversaryEvents method 34
profile_GetCurrentChangeToken method 35
profile_GetInterestsExpertsEvents method 37
profile_GetUserColleagueEvents method 33
profile_GetUserColleagueEventsByDate method 38
profile_GetUserEvents method 31
profile_GetUserEventsIDRange method 58
profile_GetUsersColleaguesAndRights method 57
profile_GetViewerPublisherInfo method 57

R

Recording a user profile change event in the activity feed example 64
References
  informative 7
  normative 7
Relationship to other protocols 9
Result sets - messages
  EnumProfileChangesOrganization 19
  EnumProfileChangesUser 16
  GetActivityApplications 20
  GetActivityEvents 20
  GetActivityPreferences 21
  GetActivityTemplates 21
  GetActivityTypes 22
  GetCurrentChangeTokenOrganization 19
  GetCurrentChangeTokenUser 17
  GetInterestsExpertsEvents 17
  GetLatestLastActivityEventUpdateTime 23
  GetUserColleagueEvents 13
  GetUserEvents 15
  GetUserEventsIDNewest 24
  GetUserEventsIDOldest 24
  GetUserEventsNoRecordId 15
  GetUserEventsRecordId 18
  GetUserInfo 23
  GetUsersColleaguesAndRights 24
Retrieve all changes for a specified user example 62
Retrieve all changes for the colleagues of a user example 61
Retrieving information from a user's consolidated feed example 70
Retrieving information from a user's published feed example 67

S

Security
  implementer considerations 73
  parameter index 73
Sequencing rules
  client 59
  server 31
Server
  abstract data model 29
  activityFeed_DeleteActivityEventsConsolidated method 38
  activityFeed_DeleteActivityEventsPublished method 39
  activityFeed_DeleteActivityTemplates method 40
  activityFeed_DeleteActivityTypes method 40
  activityFeed_DeleteIfExistsActivityPreference method 40
  activityFeed_GetActivityApplications method 41
  activityFeed_GetActivityEventsConsolidated method 42
  activityFeed_GetActivityEventsPublished method 42
  activityFeed_GetActivityPreferences method 44
  activityFeed_GetActivityTemplates method 44
  activityFeed_GetActivityTypes method 45
  activityFeed_GetLatestLastUpdateTimeConsolidated method 46
  activityFeed_GetLatestLastUpdateTimePublished method 46
  activityFeed_InsertActivityApplications method 47
  activityFeed_InsertActivityEventsConsolidated method 47
  activityFeed_InsertActivityEventsPublished method 48
  activityFeed_InsertActivityTemplates method 49

activityFeed_InsertActivityTypes method 50
activityFeed_InsertIfNotExistsActivityPreference method 52
activityFeed_UpdateActivityApplications method 52
activityFeed_UpdateActivityEventsConsolidated method 53
activityFeed_UpdateActivityEventsPublished method 54
activityFeed_UpdateActivityTemplates method 54
activityFeed_UpdateActivityTypes method 55
higher-layer triggered events 31
initialization 31
local events 58
message processing 31
profile_AddUserProfileEvent method 35
profile_DeleteUserEvents method 33
profile_EnumProfileChanges method 36
profile_GenerateAnniversaryEvents method 34
profile_GetCurrentChangeToken method 35
profile_GetInterestsExpertsEvents method 37
profile_GetUserColleagueEvents method 33
profile_GetUserColleagueEventsByDate method 38
profile_GetUserEvents method 31
profile_GetUserEventsIDRange method 58
profile_GetUsersColleaguesAndRights method 57
profile_GetViewerPublisherInfo method 57
sequencing rules 31
timer events 58
timers 31
Simple data types
Change Type 11
Object Type 11
Privacy Policy Type 11
Privacy Type 13
Value 12
Simple types - overview 25
Standards assignments 10
Structures
binary 13
table and view 25
XML 25
Synchronization example 63

T

Table structures - overview 25
Timer events
client 59
server 58
Timers
client 58
server 31
Tracking changes 75
Transport 11
Triggered events - higher-layer
client 59
server 31
Types
complex 25
simple 25

V

Value simple type 12
Vendor-extensible fields 10
Versioning 10
View structures - overview 25

X

XML structures 25