[MS-UPIESP]:
User Profile Import and Export Stored Procedures 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	1.06	Major	Significantly changed the technical content.
12/17/2010	1.06	No change	No changes to the meaning, language, or formatting of the technical content.
03/18/2011	1.06	No change	No changes to the meaning, language, or formatting of the technical content.
06/10/2011	1.06	No change	No changes to the meaning, language, or formatting of the technical content.

Table of Contents

1 Introduction

This document specifies the User Profile Import and Export Stored Procedures Protocol. This protocol is used to import and export information about users and member groups.

1.1 Glossary

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

Active Directory
directory service (DS)
GUID
LDAP

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

back-end database server
Business Data Connectivity (BDC)
distribution list
member group
membership
organizational unit
partition
request identifier
result set
return code
stored procedure
tenant
user profile
user profile store

The following terms are specific to this document:

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

1.2 References

1.2.1 Normative References

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

[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".

[MS-UPIEWS] Microsoft Corporation, "User Profile Import and Export Web Service Protocol Specification"

[MS-UPSCHNG2] Microsoft Corporation, "User Profile Change Log Stored Procedure Version 2 Protocol Specification"

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

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

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

[RFC2251] Wahl, M., Howes, T., and Kille, S., "Lightweight Directory Access Protocol (v3)", RFC 2251, December 1997, http://www.ietf.org/rfc/rfc2251.txt

1.2.2 Informative References

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

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

1.3 Protocol Overview (Synopsis)

This protocol is used to import and export **user profile** and **member group** data to and from the **user profile store**. A typical scenario for using this protocol is a synchronization application that runs at fixed intervals to keep the user profile store and an **LDAP directory service** in sync.

The protocol supports methods to retrieve all user profiles or only user profiles that have changed since a specific time. The protocol also allows importing Business Data Connectivity data for specific user profile properties for existing user profiles.

1.4 Relationship to Other Protocols

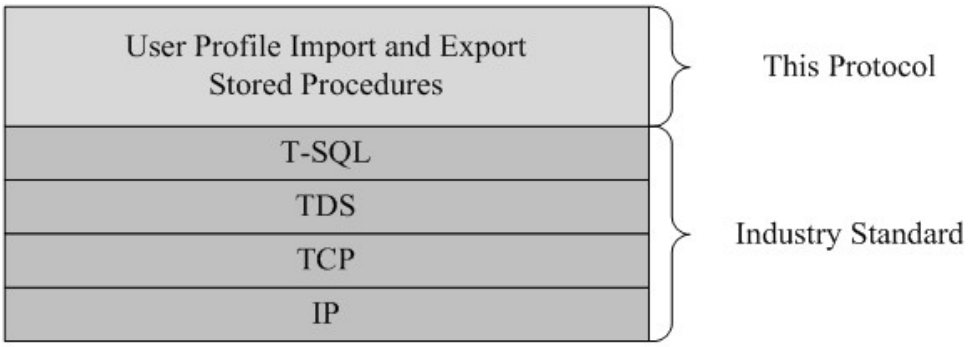

Figure 1: This protocol in relation to other protocols

1.5 Prerequisites/Preconditions

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

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

1.6 Applicability Statement

This protocol is designed for flowing user and group data across the user profile store and external directory services. It is applicable when the protocol client is acting as a broker between directory services and the user profile store.

This protocol is designed with the intention of supporting a scale point of approximately:

- 2 million users
- On average 100 member groups per user profile, up to a total of 1 million member groups
- 10 million group **memberships**

This protocol does not specify how the data should be stored in the external directory services, how the protocol client should connect to external directory services, or what synchronization logic should be used by the protocol client when flowing data between the user profile store and the external directory service.

1.7 Versioning and Capability Negotiation

Versions of the data structures or stored procedures in the database must 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] section 2 specifies the transport protocol used to call the stored procedures, query SQL tables, get **return codes**, and return **result sets**.

2.2 Common Data Types

This section contains common definitions used by this protocol.

2.2.1 Simple Data Types and Enumerations

The following simple types and enumerations are specified in this protocol.

2.2.1.1 Staging Status Type

An integer that specifies the status of the a member in the ImportExportStagedMember table (section 2.2.5.2). The value MUST be one of the values listed in the following table. If the value is not one of the values listed in the following table then the member in that row of the ImportExportStagedMember table will be ignored and will remain in the table after post import processing.

Value	Description
-1	Omit this member from import.
0	Member has not yet been processed.
1	Member has been identified as a valid user profile or member group.
2	Member post import processing is finished.

2.2.1.2 Member Type

An integer that specifies the type of member of a member group. The value MUST be one of the values listed in the following table. If the value is not one of the values listed in the following table then the member will not be imported into the user profile store.

Value	Description
0	Member type is unknown.
1	Member is a user.
2	Member is a member group.

2.2.1.3 Short Group Type

A 1-byte unsigned integer that specifies the member group type. These values are a subset of the Group Type value (section 2.2.1.4). The value MUST be one of the values listed in the following table:

Value	Description
0	User Specified grouping.
7	**Distribution list** default grouping.
8	Site default grouping.

2.2.1.4 Group Type

A 1-byte integer that specifies the member group type. This value MUST be one of the values listed in the following table:

Value	Description
0	User Specified grouping.
1	Best Bet. User specified group which has an emphasized link in the user interface.
2	General.
5	Users who share the same **Manager** property.
7	Distribution list default grouping.
8	Site default grouping.

Values 3, 4, and 6 are undefined.

2.2.1.5 Is Expanded Type

A bit specifying whether the relation was added as a result of expanding the members of groups within a group. This value MUST be one of the values listed in the following table. If a value is used which is not in the following table then the behavior is undefined.

Value	Description
0	The user is a member of the group.
1	This user is a member of a subgroup of the group.

2.2.1.6 Short Link Type

A **GUID** that specifies the source of a member group. This value MUST be listed in the following table. If a value is used which is not in the following table then the behavior is undefined.

Value	Description
A88B9DCB-5B82-41E4-8A19-17672F307B95	Specifies a member group which is a distribution list.
8BB1220F-DE8B-4771-AC3A-0551242CF2BD	Specifies a site sourced member group.

2.2.2 Bit Fields and Flag Structures

None.

2.2.3 Binary Structures

None.

2.2.4 Result Sets

2.2.4.1 ImportExport_GetGroupMembers.ResultSet0

This result set returns the distinguished names (1) of the users and groups which are members of a group. The ImportExport_GetGroupMembers.ResultSet0 MUST contain 0 rows when the *@GroupId* input parameter does not specify a valid group.

```
DistinguishedName nvarchar(2048),
```

DistinguishedName: A string compatible with the LDAP standard distinguished name (1), see [RFC2251]. This string specifies the distinguished name (1) of a member in a member group.

2.2.4.2 ImportExport_GetNonimportedObjects.ResultSet0

This result set returns the RecordID, PartitionID and NTName of the user profiles that have not been imported. A user profile is recognized as imported when the user profile has a corresponding row in the DNLookup table ([MS-UPSPROF2] section 2.2.5.1). An imported user profile has a distinguished name (1) stored in the DNLookup table ([MS-UPSPROF2] section 2.2.5.1) which is the identifier used by the data source from which the user profile is imported. User profiles added to the profile store by means other than import do not have a corresponding row in the DNLookup table ([MS-UPSPROF2] section 2.2.5.1).

```
RecordID bigint,
PartitionID uniqueidentifier,
NTName nvarchar(400),
```

RecordID: RecordId of the user profile as specified in the UserProfile_Full table ([MS-UPSPROF2] section 2.2.5.4).

PartitionID: PartitionID of the user profile as specified in the UserProfile_Full table ([MS-UPSPROF2] section 2.2.5.4).

NTName: NTName of the user profile as specified in the UserProfile_Full table ([MS-UPSPROF2] section 2.2.5.4).

2.2.4.3 ImportExport_GetNonimportedObjects.ResultSet1

This result set returns the Id, PartitionID and SourceReference of the member groups that are not imported. A member group is recognized as imported when the member group has a corresponding row in the DNLookup table ([MS-UPSPROF2] section 2.2.5.1). An imported member group has a distinguished name (1) stored in the DNLookup table ([MS-UPSPROF2] section 2.2.5.1) which is the identifier used by the data source from which the member group is imported. Member groups added to the profile store by means other than import do not have a corresponding row in the DNLookup table ([MS-UPSPROF2] section 2.2.5.1).

```
Id bigint,
PartitionID uniqueidentifier,
```

```
SourceReference nvarchar(2048),
```

Id: Id of the member group as specified in the MemberGroup table ([MS-UPSPROF2] section 2.2.5.5).

PartitionID: PartitionID of the member group as specified in the MemberGroup table ([MS-UPSPROF2] section 2.2.5.5).

SourceReference: SourceReference of the member group as specified in the MemberGroup table ([MS-UPSPROF2] section 2.2.5.5).

2.2.5 Tables and Views

The following tables are referenced in this protocol. No views are referenced in this protocol.

2.2.5.1 ImportExport

The ImportExport table is used to track the batches of users and member groups being imported. When an import batch is started a row MUST be created in this table. The ImportExportId of that new row MUST be passed as the ImportExportId parameter in subsequent calls to perform import actions. A new row SHOULD NOT be created when exporting data. Creating a new row when exporting data has no effect and the row will be ignored.

```
ImportExportId bigint NOT NULL,
StartTime datetime NOT NULL,
EndTime datetime NULL,
IsImport bit NOT NULL,
```

ImportExportId: A GUID which uniquely identifies the import batch.

StartTime: The starting date and time of the import batch.

EndTime: The ending date and time of the import batch. This value MUST be NULL until the import process is complete. When the import process is complete, then this value MUST be set to a valid date. If the value is set before the import process is complete then the ImportExport_IsRunning stored procedure (Section 3.1.5.10) may incorrectly return false during the import process. The EndTime SHOULD be greater than the StartTime column. If the EndTime is equal to or less than the StartTime then the behavior is undefined.

IsImport: A bit value indicating if this batch is an import or an export. This bit SHOULD be set to 1 when starting an import process and 0 for starting an export process.

2.2.5.2 ImportExportStagedMember

This table is used to temporarily store the members of a member group, which can include both users and other member groups.

```
ImportExportId bigint NOT NULL,
StagedId bigint NOT NULL,
MemberDN nvarchar(max) NOT NULL,
MemberId bigint NULL,
MemberType int NULL,
ParentGroupId bigint NOT NULL,
```

```
Status int NOT NULL,
```

ImportExportId: A GUID specifying the import batch associated with this imported member record. This ImportExportId MUST correspond to a row in the ImportExport table (section 2.2.5.1).

StagedId: A 64-bit integer which uniquely identifies a row of data in this table.

MemberDN: A string containing the LDAP standard distinguished name [RFC2251] of the member.

MemberId: A 64-bit integer which identifies the member.

MemberType: Contains a Member Type (section 2.2.1.2) value which specifies the type of member.

ParentGroupId: A 64-bit integer which identifies the member group which contains the member.

Status: Contains a Staging Status Type (section 2.2.1.1) which specifies the status of the member in during the post import processing.

2.2.5.3 ProfileImportStagingPersonProperties

This table is used to temporarily store the properties associated with a user profile.

```
PartitionID uniqueidentifier NOT NULL,
Id bigint NOT NULL,
RecordId bigint NOT NULL,
ProfileType nvarchar(20) NULL,
PropertyId bigint NULL,
PropertyVal sql_variant NULL,
```

PartitionID: A globally unique identifier (GUID) used to filter the current request. This value MUST NOT be null or empty.

Id: A 64-bit integer which uniquely identifies a row of data in this table.

RecordId: A 64-bit integer identifier of the user profile which this property is associated with.

ProfileType: Type the Column description.

PropertyId: A 64-bit integer identifier of the property being associated with the user profile. This value MUST have a corresponding row in the PropertyList table ([MS-UPSCHNG2] section 3.1.1).

PropertyVal: A variant containing the data for the property. This value MUST be of the type specified for the property in the PropertyList table.

2.2.6 XML Structures

This section describes the XML schema used in this protocol.

2.2.6.1 Members XML

The Members XML structure MUST be used for the @members parameter of the ImportExport_ImportMembers stored procedure.

The Members XML is an XML fragment which MUST contain one and only one <Ms> element and SHOULD contain one or more <M> elements.

The <Ms> element is the root element of the XML fragment. If the XML fragment contains additional <Ms> elements at the root level, then an error will occur when attempting to read the XML fragment. Additional <Ms> elements under the root <Ms> are ignored. If no <M> element is specified then the XML fragment is ignored because it contains no member data to process.

2.2.6.2 Namespaces

None.

2.2.6.3 Simple Types

None.

2.2.6.4 Complex Types

None.

2.2.6.5 Elements

The following are XML elements specified in this protocol:

Element	Description
Ms	Root element of a list of members of a group.
M	Element which specifies a member of a group. The parent of this element MUST be the <Ms> element. The <M> element MUST have a DN and an OU attribute.

2.2.6.6 Attributes

The following are XML attributes specified in this protocol:

Attribute	Description
DN	The distinguished name (1) of the member.
OU	The **organizational unit** of the member.

2.2.6.7 Groups

None.

2.2.6.8 Attribute Groups

None.

3 Protocol Details

3.1 Server Details

The back-end database protocol responds to stored procedure calls. It returns result sets and return codes and never initiates communication with other endpoints.

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.

The protocol server maintains the following data:

- A list of member groups and the user profiles that belong to them.
- A reconciliation of member groups that contain other member groups such that users belonging to a child member group are identified as belonging in the parent member group.
- A list of partitions (1).
- A list of user profile properties and the values that belong to them.
- The state information about whether a profile import or export session is in progress.

The following diagram illustrates the relationships between the tables referenced in this protocol:

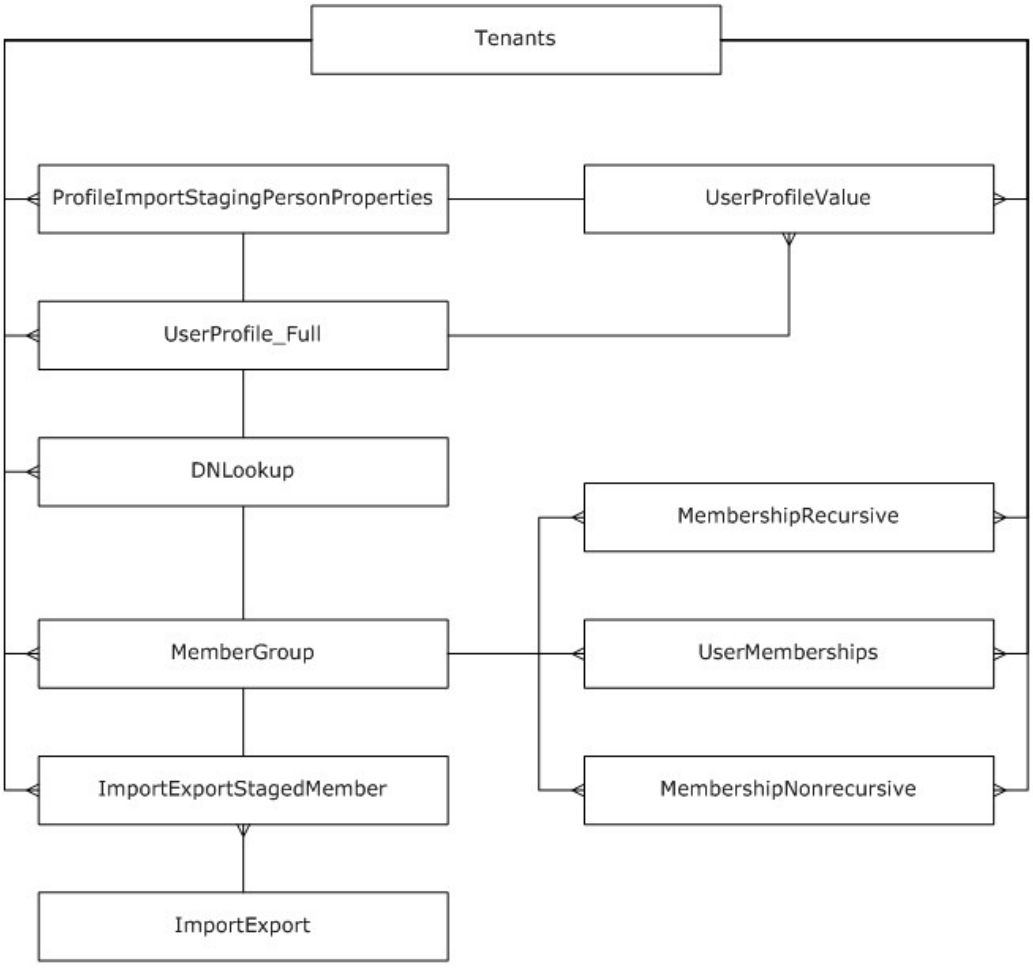

Figure 2: Abstract Data Model

The UserProfile_Full ([MS-UPSPROF2] section 2.2.5.4), MemberGroup ([MS-UPSPROF2] section 2.2.5.5), and DNLookup ([MS-UPSPROF2] section 2.2.5.2) tables are expected to contain existing users and member groups. This protocol's import process loads member group members into the ImportExportStagedMember (section 2.2.5.2) table and user profile properties into the ProfileImportStagingPersonProperties (section 2.2.5.3) table through the ImportExport_ImportMembers (section 3.1.5.5) and profile_UpdateStagingPersonProperty (section 3.1.5.13) stored procedures respectively.

Once the import of data is finished, ImportExport_PostImportMembers (section 3.1.5.8) and ImportExport_PostImportUserProperties (section 3.1.5.9) stored procedures are called to update the member group members and user profile properties in the operational tables. ImportExport_PostImportMembers (section 3.1.5.8) updates the membership data in the MembershipRecursive ([MS-UPSPROF2] section 2.2.5.6), MembershipNonRecursive([MS-UPSPROF2] section 2.2.5.5), and UserMemberships ([MS-UPSPROF2] section 2.2.5.6) tables. The ImportExport_PostImportUserProperties (section 3.1.5.9) stored procedure updates the user profile attributes in the UserProfileValue ([MS-UPSPROF2] section 2.2.5.4) and UserProfile_Full ([MS-UPSPROF2] section 2.2.5.4) tables.

The ImportExport_GetGroupMembers (section 3.1.5.12) stored procedure retrieves the groups from the operational tables MembershipNonrecursive ([MS-UPSPROF2] section 2.2.5.5) and UserMemberships ([MS-UPSPROF2] section 2.2.5.6).

The Tenants table contains the **partition** (1) identifier which is a foreign key in all tables except the ImportExport table. The ImportExport_GetPartitionId (section 3.1.5.11) retrieves the **tenant** partition (1) identifier from the Tenants table ([MS-UPSPROF2] section 2.2.5.7).

3.1.2 Timers

None.

3.1.3 Initialization

When performing an import of user profiles and member groups, the ImportExport_ImportStart (section 3.1.5.7) stored procedure MUST be the first stored procedure called.

3.1.4 Higher-Layer Triggered Events

None.

3.1.5 Message Processing Events and Sequencing Rules

The following stored procedures MUST be called in the order shown when importing user and group data. The call sequence for ImportExport_PostImportUserProperties (section 3.1.5.9) and ImportExport_PostImportMembers (section 3.1.5.8) MUST be called after ImportExport_ImportEnd (section 3.1.5.6). If these stored procedures are called before the import batch has finished, then errors may occur due the import data lack of referential integrity.

1. ImportExport_ImportStart (section 3.1.5.7)

2. ImportExport_ImportMembers (section 3.1.5.5)

3. ImportExport_ImportEnd (section 3.1.5.6)

4. ImportExport_PostImportUserProperties (section 3.1.5.9) and ImportExport_PostImportMembers (section 3.1.5.8).

Only one import batch MUST be in process at a time. The ImportExport_PostImportUserProperties and ImportExport_PostImportMembers stored procedures MUST be after the end of the data import, which is signified by the call to the ImportExport_ImportEnd stored procedure, and MUST be called before additional another import is started. If ImportExport_PostImportUserProperties and ImportExport_PostImportMembers are called while data is being imported the stored procedures MAY fail because of the lack of referential integrity in the import data.

3.1.5.1 profile_GetBusinessDataCatalogConnections

This stored procedure returns the list of **BDC** Profile Synchronization connections.

```
PROCEDURE DBO.profile_GetBusinessDataCatalogConnections (
@partitionID uniqueidentifier
,@correlationId uniqueidentifier = NULL
);
```

@partitionID: A GUID which identifies the tenant partition (1) identifier of the member group which the members are being added.

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

Return Values: MUST NOT return any values.

Result Sets: This stored procedure MUST return one result set.

3.1.5.1.1 profile_GetBusinessDataCatalogConnections Result Set

This result set MUST return 0 or more rows. For a record to be included in the result set, it MUST NOT have a NULL display name.

The profile_EnumUsers result set is defined using T-SQL syntax as follows:

```
DisplayName                string,
SystemName                 string,
EntityName                 string,
EntityNamespace            string,
FilterName                 string,
ProfilePropertyName        string,
MappedAttribute             string;
```

DisplayName: Name of the Profile Synchronization Connection.

SystemName: Name of the BDC System for Synchronize data from.

EntityName: Name of the BDC Entity for Synchronize data from.

EntityNamespace: NameSpace of the BDC Entity for Synchronize data from.

FilterName: Name of the BDC filter to implement on Profile synchronization.

ProfilePropertyName: Name of the User Profile property used to identify the User profile to add BDC data to.

MappedAttribute: Name of the BDC entity attribute whose value should match the ProfileProperty value for a successful join to happen.

3.1.5.2 profile_EnumerateUsersForBDCImport

This stored procedure returns the all the imported user profiles with the property value for the property that is set as the Join attribute for a BDC import

```
PROCEDURE [dbo].[profile_EnumerateUsersForBDCImport] (
@partitionID uniqueidentifier
,@mossJoinAttribute nvarchar(250)
,@beginID bigint
,@pageSize int
,@correlationId uniqueidentifier = NULL
);
```

@partitionID: A GUID which identifies the tenant partition (1) identifier of the member group which the members are being added.

@mossJoinAttribute: Name of the User Profile Property whose value is used to identify the row from BDC entity data which is used to add data to mapped profile properties.

@beginID: The value where the Protocol server starts searching for existing user profile record identifiers. This parameter MUST be specified and it MUST NOT be NULL.

@pageSize The value which determined the number of user profiles to retrieve.

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

Return Values: MUST NOT return any values.

Result Sets: This stored procedure MUST return one result set.

3.1.5.2.1 profile_EnumerateUsersForBDCImport Result Set

This result set returns a page full of imported users present in the profile store with their distinguished name. The result set MUST contain at-maximum @pageSize number of rows.

DNId: A unique identifier associated with the distinguished name (1).

RecordId: RecordId of the user profile as present in the profile store.

MossJoinValue: The value the user profile property passed in as @mossJoinAttribute.

DN: A string compatible with the LDAP standard distinguished name (1).

3.1.5.3 profile_DeleteBusinessDataCatalogConnection

This stored procedure deletes the Profile Synchronization connection for the BusinessDataCatalog with the given displayName.

```
PROCEDURE DBO.profile_DeleteBusinessDataCatalogConnection (
@partitionID uniqueidentifier
,@displayName nvarchar(128)
,@correlationId uniqueidentifier = NULL
);
```

@partitionID: A GUID which identifies the tenant partition (1) identifier of the member group which the members are being added.

@displayName: Name of Profile Synchronization connection.

@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.

3.1.5.4 profile_UpdateBusinessDataCatalogConnection

This stored procedure updates and creates the Profile Synchronization connection for the BusinessDataCatalog.

```
PROCEDURE DBO.profile_UpdateBusinessDataCatalogConnection (
@partitionID uniqueidentifier
```

```
,@correlationId uniqueidentifier = NULL
,@displayName nvarchar(128)
,@systemName nvarchar(250)
,@entityName nvarchar(250)
,@entityNamespace nvarchar(250)
,@filterName nvarchar(250)
,@profilePropertyName nvarchar(50,
,@mappedAttribute nvarchar(250)
);
```

@partitionID: A GUID which identifies the tenant partition (1) identifier of the member group which the members are being added.

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

@displayName: Name of the Profile Synchronization Connection.

@systemName: Name of the BDC System for Synchronize data from.

@entityName: Name of the BDC Entity for Synchronize data from.

@entityNamespace: NameSpace of the BDC Entity for Synchronize data from.

@filterName: Name of the BDC filter to implement on Profile synchronization.

@profilePropertyName: Name of the User Profile property used to identify the User profile to add BDC data to.

@mappedAttribute: Name of the BDC entity attribute whose value should match the ProfileProperty value for a successful join to happen.

Return Values: An integer which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.1.5.5 ImportExport_ImportMembers

This stored procedure stores the members of a group in the ImportExportStagedMember table (section 2.2.5.2) during the import process. After the import process has finished, the stored procedure ImportExport_PostImportMembers (section 3.1.5.8) MUST be called to complete the transference of imported members to the operational tables of the user profile store. If this stored procedure fails then none of the group members will be stored in the ImportExportStagedMember table.

```
PROCEDURE ImportExport_ImportMembers (
@importExportId bigint
,@members nvarchar(max)
,@parentGroupId bigint
,@partitionID uniqueidentifier
,@correlationId uniqueidentifier = null
);
```

@importExportId: The 64-bit integer identifier of the import or export batch being processed. This value MUST be the value of the @importExportId output parameter of the last call to the ImportExport_ImportStart (section 3.1.5.7) stored procedure.

@members: A Members XML (section 2.2.6.1) string which specifies the members of the member group.

@parentGroupId: The 64-bit identifier that identifies the member group which contains the members which are being added.

@partitionID: A GUID which identifies the tenant partition (1) identifier of the member group which the members are being added.

@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.

3.1.5.6 ImportExport_ImportEnd

This stored procedure is called to signify the completion of the specified import batch. If this stored procedure fails, then the import batch is not updated to signify completion of the import batch.

```
PROCEDURE ImportExport_ImportEnd (
@importExportId bigint
,@correlationId uniqueidentifier = null
);
```

@importExportId: The 64-bit integer identifier of the import or export batch being processed. This value MUST be the value of the @importExportId output parameter of the last call to the ImportExport_ImportStart (section 3.1.5.7) stored procedure. Supplying a value other than the value corresponding call to ImportExport_ImportStart will result in error.

@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.

3.1.5.7 ImportExport_ImportStart

This stored procedure is called prior to importing a batch of users and groups. The importExportId output by this stored procedure is used in subsequent calls to stored procedures in this protocol. When the import has finished, a call to the stored procedure ImportExport_ImportEnd (section 3.1.5.6) MUST be made using the importExportId output from this stored procedure. Only one import batch MUST be processed at a time, so this stored procedure MUST NOT be called more than once before the call to ImportExport_ImportEnd (section 3.1.5.6). If this stored procedure is called more than once before ImportExport_ImportEnd it will succeed, however the state of the import process will then be undefined. If this stored procedure fails then the importExportId that is output will not contain a valid identifier.

```
PROCEDURE ImportExport_ImportStart (
@importExportId bigint OUTPUT
,@correlationId uniqueidentifier = null
);
```

@importExportId: A 64-bit integer identifier used to represent the import being started.

Value	Description
@@identity	The importExportId output is the value of @@identity after calling this stored procedure.

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

Return Values: An integer which MUST be zero.

Value	Description
0	Default return value.

Result Sets: MUST NOT return any result sets.

3.1.5.8 ImportExport_PostImportMembers

This stored procedure is called to process the member information in the ImportExportStagedMembers table (section 2.2.5.2) and MUST be called after the completion of the import batch to ensure the imported data has referential integrity. This stored procedure MUST only be run after the call to ImportExport_ImportEnd (section 3.1.5.6) has been made and before the next import is started. If this stored procedure fails, then all the post processing work may not have been finished and the stored procedure MUST be called again to complete the post processing of the imported members.

```
PROCEDURE ImportExport_PostImportMembers (
@correlationId uniqueidentifier = null
);
```

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

Return Values: An integer which MUST be 0 on success and nonzero on failure.

Result Sets: MUST NOT return any result sets.

3.1.5.9 ImportExport_PostImportUserProperties

This stored procedure is called to process the user profile properties which were saved to the ProfileImportStagingPersonProperties table (section 2.2.5.3) during the user profile import. This stored procedure MUST be called after the completion of the import batch to ensure the imported data has referential integrity. This stored procedure MUST only be run after the call to ImportExport_ImportEnd (section 3.1.5.6) has been made and before the next import is started.

```
PROCEDURE ImportExport_PostImportUserProperties (
@correlationId uniqueidentifier = null
);
```

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

Return Values: MUST NOT return any values.

Result Sets: MUST NOT return any result sets.

3.1.5.10 ImportExport_IsRunning

This stored procedure is invoked to determine if an import or export is currently running.

```
PROCEDURE ImportExport_IsRunning (
@correlationId uniqueidentifier = null
);
```

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

Return Values: An integer which MUST be zero when no import or export is currently running or MUST be one when an import or export is running.

Result Sets: MUST NOT return any result sets.

3.1.5.11 ImportExport_GetPartitionId

This stored procedure retrieves the partition (1) identifier of the tenant with the corresponding organizational unit when multiple tenants exist or the default partition (1) identifier for a single tenant. If this stored procedure fails then the partition (1) identifier output is undefined and MUST not be used.

```
PROCEDURE ImportExport_GetPartitionId (
@organizationalUnit nvarchar(64)
,@correlationId uniqueidentifier = null
,@partitionId uniqueidentifier OUTPUT
);
```

@organizationalUnit: The name of the organizational unit that corresponds to the partition (1) identifier. When multiple tenants exist, this value MUST correspond to a value in the SynchronizationOU column in the Tenants table ([MS-UPSPROF2] section 2.2.5.7).

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

@partitionId: A globally unique identifier (GUID) used to filter the current request. This value MUST NOT be null or empty.

Value	Description
0	Default return value.

Return Values: An integer which MUST be zero.

Result Sets: MUST NOT return any result sets.

3.1.5.12 ImportExport_GetGroupMembers

This stored procedure return a result set with the distinguished name (1) of all the members of the specified member group from the operational user profile store. The distinguished names (1) of both user profiles and member group members are returned in the result set.

```
PROCEDURE ImportExport_GetGroupMembers (
@partitionID uniqueidentifier
,@Id bigint
```

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

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

@Id: The identifier of the member group whose members are to be retrieved.

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

Return Values: An integer which MUST be zero.

Result Sets:

This stored procedure MUST return a ImportExport_GetGroupMembers.ResultSet0

3.1.5.13 profile_UpdateStagingPersonProperty

This stored procedure stores the user profile properties in the ProfileImportStagingPersonProperties table (section 2.2.5.3) during the import process. After the import processes has finished, the stored procedure ImportExport_PostImportUserProperties (section 3.1.5.9) MUST be called to transfer the members to the operational tables of the user profile store.

```
PROCEDURE profile_UpdateStagingPersonProperty (
@partitionID uniqueidentifier
,@RecordId bigint
,@ProfileType nvarchar(20)
,@PropertyURI nvarchar(250)
,@PropertyVal sql_variant
,@correlationId uniqueidentifier = null
);
```

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

@RecordId: An integer that specifies the user profile which the properties are to be updated.

@ProfileType: A string that specifies the type of profile to be updated. The values for this parameter MUST be either "UserProfile" or "OrganizationalProfile". Use of other values will result in errors.

@PropertyURI: A string which contains the name that identifies the user profile property.

@PropertyVal: The value of the user profile property.

@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.

3.1.5.14 ImportExport_CleanGroupMembers

This stored procedure removes all member groups that are members of the specified member group from the MembershipRecursive ([MS-UPSPROF2] section 2.2.5.6) and MembershipNonRecursive ([MS-UPSPROF2] section 2.2.5.5) tables.

```
PROCEDURE ImportExport_CleanGroupMembers (
@memberGroupId bigint
,@partitionId uniqueidentifier
,@correlationId uniqueidentifier = null
);
```

@memberGroupId: The identifier of the member group which contains the members which are to be deleted.

@partitionId: A globally unique identifier (GUID) 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.

3.1.5.15 ImportExport_PurgeNonimportedObjects

This stored procedure will purge the user profiles and member groups that are not imported using this protocol (section 1.3) nor using MS-UPIEWS protocol ([MS-UPIEWS] section 1.3).

For user profiles, the stored procedure marks all the user profiles in the UserProfile_Full table ([MS-UPSPROF2] section 2.2.5.4) that do not have a corresponding row in the DNLookup table ([MS-UPSPROF2] section 2.2.5.2) as deleted.

For member groups, the stored procedure deletes all the member groups in the MemberGroup table ([MS-UPSPROF2] section 2.2.5.5) that do not have a corresponding row in the DNLookup table ([MS-UPSPROF2] section 2.2.5.2).

```
PROCEDURE ImportExport_PurgeNonimportedObjects (
@isUsersOnly bit = null
,@correlationId uniqueidentifier = null
);
```

@isUsersOnly: Specifies if the operation is performed for user profiles only, or for both user profiles and member groups. If this value is NULL or zero, the operation MUST be performed for both user profiles and member groups. For all other values the operation MUST be performed on user profiles only and member groups MUST remain unchanged.

@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.

3.1.5.16 ImportExport_GetNonimportedObjects

This stored procedure returns result sets containing the user profiles and member groups that are not imported using this protocol (section 1.3) nor using MS-UPIEWS protocol ([MS-UPIEWS] section 1.3).

For user profiles, the stored procedure returns a ImportExport_GetNonimportedObjects.ResultSet0 containing all the user profiles in the UserProfile_Full table ([MS-UPSPROF2] section 2.2.5.4) that do not have a corresponding row in the DNLookup table ([MS-UPSPROF2] section 2.2.5.2).

For member groups, the stored procedure returns a ImportExport_GetNonimportedObjects.ResultSet1 containing all the member groups in the MemberGroup table ([MS-UPSPROF2] section 2.2.5.5) that do not have a corresponding row in the DNLookup table ([MS-UPSPROF2] section 2.2.5.2). This result set MUST be returned only when @isUsersOnly is NULL or zero.

```
PROCEDURE ImportExport_GetNonimportedObjects (
@isUsersOnly bit = null
,@correlationId uniqueidentifier = null
);
```

@isUsersOnly: Specifies if the operation returns the records for user profiles only, or for both user profiles and member groups. If this value is NULL or zero, the operation MUST return both user profile and member groups result sets. For all other values the operation MUST return the user profiles result set only and MUST NOT return a result set for member groups.

@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 ImportExport_GetNonimportedObjects.ResultSet0

If and only if @isUsersOnly is NULL or zero, this stored procedure MUST also return a ImportExport_GetNonimportedObjects.ResultSet1

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 stored procedures are used to import a single group with 5 users from **Active Directory**. After the import has finished, the imported member group members are added to the user profile store.

```
-- Sample value for default partition identifier
DECLARE @partitionId   uniqueidentifier SET @partitionId = '0c37852b-34d0-418e-91c6-
2ac25af4be5b'
DECLARE @correlationId   uniqueidentifier SET @partitionId = '00000000-0000-0000-0000-
000000000000'

-- Sample member group values
DECLARE @parentGroupId bigint           SET @parentGroupId = 100

-- Sample XML structure passed to the procedure for ImportExport_ImportMembers
DECLARE @members nvarchar(max)
SET @members = '
<Ms>
    <M DN="CN=UserOne,OU=UserAccounts,DC=DOMAINNAME,DC=corp,DC=COMPANYNAME,DC=com", OU=
"UserAccounts" />
    <M DN="CN=UserTwo,OU=UserAccounts,DC=DOMAINNAME,DC=corp,DC=COMPANYNAME,DC=com", OU=
"UserAccounts" />
    <M DN="CN=UserThree,OU=UserAccounts,DC=DOMAINNAME,DC=corp,DC=COMPANYNAME,DC=com", OU=
"UserAccounts" />
    <M DN="CN=UserFour,OU=UserAccounts,DC=DOMAINNAME,DC=corp,DC=COMPANYNAME,DC=com", OU=
"UserAccounts" />
    <M DN="CN=UserFive,OU=UserAccounts,DC=DOMAINNAME,DC=corp,DC=COMPANYNAME,DC=com", OU=
"UserAccounts" />
</Ms>'

-- Start the import process
DECLARE @ImportExportId uniqueidentifier
EXEC @ImportExportId = [dbo].[ImportExport_ImportStart] @correlationId
```

OUTPUT: After successful execution of this stored procedure, it returns 1 (sample data) which is assigned to importExportId.

```
-- Store the members temporarily in SQL tables for later processing
EXEC [dbo].[ImportExport_ImportMembers] @ImportExportId, @members, @parentGroupId,
@partitionId, @correlationId
```

OUTPUT: After successful execution of this stored procedure, the sample data in the table will be as shown here.

ImportExportId	MemberDN	MemberOU	ParentGroupDN	ParentGroupID
1	CN=UserOne,OU=UserAccounts,DC=DOMAINNAME,DC=corp,DC=COMPANYNAME,DC=com	UserAccounts	CN=USERNAME,OU=Distribution Lists,DC=DOMAINNAME,DC=corp,DC=COMPANYNAME,DC=com	100
1	CN=UserTwo,OU=UserAccounts,DC=DOMAI	UserA	CN=USERNAME,OU=Distributi	100

ImportExportId	MemberDN	MemberOU	ParentGroupDN	ParentGroupID
	NNAME,DC=corp,DC=COMPANYNAME,DC=com	ccounts	on Lists,DC=DOMAINNAME,DC=corp,DC=COMPANYNAME,DC=com	
1	CN=UserThree,OU=UserAccounts,DC=DOMAINNAME,DC=corp,DC=COMPANYNAME,DC=com	UserAccounts	CN=USERNAME,OU=Distribution Lists,DC=DOMAINNAME,DC=corp,DC=COMPANYNAME,DC=com	100
1	CN=UserFour,OU=UserAccounts,DC=DOMAINNAME,DC=corp,DC=COMPANYNAME,DC=com	UserAccounts	CN=USERNAME,OU=Distribution Lists,DC=DOMAINNAME,DC=corp,DC=COMPANYNAME,DC=com	100
1	CN=UserFive,OU=UserAccounts,DC=DOMAINNAME,DC=corp,DC=COMPANYNAME,DC=com	UserAccounts	CN=USERNAME,OU=Distribution Lists,DC=DOMAINNAME,DC=corp,DC=COMPANYNAME,DC=com	100

```
-- End the import process
EXEC [dbo].[ImportExport_ImportEnd] @ImportExportId, @correlationId

-- Perform the post processing of temporarily stored members
DECLARE @PostImportUserPropRet uniqueidentifier
EXEC @PostImportUserPropRet = [dbo].[ImportExport_PostImportUserProperties]
```

OUTPUT: After successful execution of this stored procedure, it returns 0 which is assigned to PostImportUserPropRet.

```
DECLARE @PostImportUserPropRet uniqueidentifier
EXEC @PostImportMemPropRet = [dbo].[ImportExport_PostImportMembers] @correlationId
```

OUTPUT: After successful execution of this stored procedure, it returns 0 which is assigned to PostImportMemPropRet.

```
-- Confirm that the members were successfully imported
-- The result set from ImportExport_GetGroupMembers displays the
-- distinguished names of the members in the member group.
EXEC [dbo].[ImportExport_GetGroupMembers] @parentPartitionId, @parentGroupId, @correlationId
```

OUTPUT: The output after successful execution of this stored procedure is as follows:

DistinguishedName
CN=UserOne,OU=UserAccounts,DC=DOMAINNAME,DC=corp,DC=COMPANYNAME,DC=com
CN=UserTwo,OU=UserAccounts,DC=DOMAINNAME,DC=corp,DC=COMPANYNAME,DC=com

DistinguishedName
CN=UserThree,OU=UserAccounts,DC=DOMAINNAME,DC=corp,DC=COMPANYNAME,DC=com
CN=UserFour,OU=UserAccounts,DC=DOMAINNAME,DC=corp,DC=COMPANYNAME,DC=com
CN=UserFive,OU=UserAccounts,DC=DOMAINNAME,DC=corp,DC=COMPANYNAME,DC=com

5 Security

5.1 Security Considerations for Implementers

None.

5.2 Index of Security Parameters

None.

6 Appendix A: Product Behavior

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include released service packs:

- Microsoft® 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

N

O

P

R

S

T

V

X