[MS-UPSCWS]:
User Profile Service Application Caching Web Service Protocol Specification

---

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	1.05	No change	No changes to the meaning, language, or formatting of the technical content.
03/18/2011	1.05	No change	No changes to the meaning, language, or formatting of the technical content.
06/10/2011	1.05	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 Service Application Caching Web Service Protocol, which enables a protocol client to retrieve user information from a database that stores user profile data for a site.

1.1 Glossary

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

authentication
Coordinated Universal Time (UTC)
domain
GUID
Hypertext Transfer Protocol (HTTP)
Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS)
security identifier (SID)
Transmission Control Protocol (TCP)
Unicode

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

application server
back-end database server
colleague
e-mail address
endpoint
login name
request identifier
Session Initiation Protocol (SIP) address
Simple Object Access Protocol (SOAP)
SOAP action
SOAP body
SOAP fault
stream
Uniform Resource Identifier (URI)
user profile
user profile record identifier
User Profile Service
user profile store
Web Services Description Language (WSDL)
WSDL message
WSDL operation

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.

[MS-SPSTWS] Microsoft Corporation, "SharePoint Security Token Service Web Service Protocol Specification"

[MS-UPSCSP] Microsoft Corporation, "User Profile Service Application Caching Stored Procedures Protocol Specification"

[MS-UPWCFWS] Microsoft Corporation, "User Profile Property Service Application Web Service 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

[RFC2616] Fielding, R., Gettys, J., Mogul, J., et al., "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999, http://www.ietf.org/rfc/rfc2616.txt

[SOAP1.1] Box, D., Ehnebuske, D., Kakivaya, G., et al., "Simple Object Access Protocol (SOAP) 1.1", May 2000, http://www.w3.org/TR/2000/NOTE-SOAP-20000508/

[SOAP1.2/1] Gudgin, M., Hadley, M., Mendelsohn, N., Moreau, J., and Nielsen, H.F., "SOAP Version 1.2 Part 1: Messaging Framework", W3C Recommendation, June 2003, http://www.w3.org/TR/2003/REC-soap12-part1-20030624

[WSDL] Christensen, E., Curbera, F., Meredith, G., and Weerawarana, S., "Web Services Description Language (WSDL) 1.1", W3C Note, March 2001, http://www.w3.org/TR/2001/NOTE-wsdl-20010315

[XMLNS] Bray, T., Hollander, D., Layman, A., et al., Eds., "Namespaces in XML 1.0 (Third Edition)", W3C Recommendation, December 2009, http://www.w3.org/TR/2009/REC-xml-names-20091208/

[XMLSCHEMA] World Wide Web Consortium, "XML Schema", September 2005, http://www.w3.org/2001/XMLSchema

[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".

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

[MS-SPTWS] Microsoft Corporation, "Service Platform Topology Web Service Protocol Specification"

[RFC2822] Resnick, P., Ed., "Internet Message Format", STD 11, RFC 2822, April 2001, http://www.ietf.org/rfc/rfc2822.txt

[SOAP1.2/2] Gudgin, M., Hadley, M., Mendelsohn, N., Moreau, J., and Nielsen, H.F., "SOAP Version 1.2 Part 2: Adjuncts", W3C Recommendation, June 2003, http://www.w3.org/TR/2003/REC-soap12-part2-20030624

1.3 Protocol Overview (Synopsis)

This protocol allows protocol clients to gain access to **user profile** information through a middle-tier **application server**, instead of a **back-end database server**. A typical scenario is a protocol client fetching user profile properties, such as the name, **e-mail address**, picture URL, or **colleagues** of one or more users, by using user credentials, such as an user name or **security identifier (SID)**, or another field, such as the **user profile record identifier**, that identify those users. The middle-tier application server caches user profile fields that are commonly used and exposes them to protocol clients through this protocol. Consequently, this protocol helps distribute the load from the database tier to the application tier; it helps the protocol server scale to handle higher loads because read requests for commonly used fields are the operations performed most frequently. In addition, this protocol is designed for bulk operations. Protocol clients can use this protocol to gain access to user profile information for many users simultaneously, and the corresponding response can contain data for as many users as requested. This feature optimizes the efficiency of operations for a large user base.

1.4 Relationship to Other Protocols

This protocol uses the **SOAP** message protocol for formatting request and response messages, as described in [SOAP1.1], [SOAP1.2/1] and [SOAP1.2/2]. It transmits those messages by using **HTTP**, as described in [RFC2616].

The User Profile Service Application Caching Web Service Protocol uses SOAP over HTTP(S) as shown in the following layering diagram:

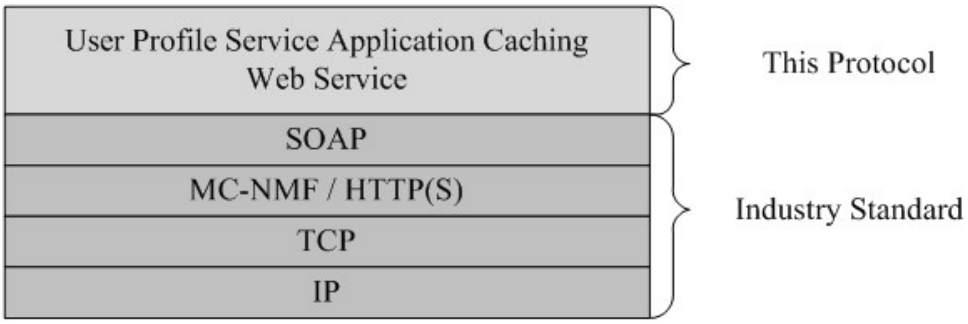

Figure 1: This protocol in relation to other protocols

1.5 Prerequisites/Preconditions

This protocol operates against a protocol server that exposes one or more **endpoint(4) URIs** that are known by protocol clients. Both the protocol server endpoint (4) URI and transport are either known by the protocol client or are obtained by using the discovery mechanism described in [MS-SPTWS].

The protocol client can obtain the **ApplicationClassId** and **ApplicationVersion** of the protocol server with which it communicates. The endpoint (4) URI of a protocol server is required to provide the discovery mechanism described in [MS-SPTWS] .

This protocol requires the protocol client to have the necessary permission settings to call methods on the protocol server. The protocol also requires that the protocol client implements the token-based security needed for the protocol server and the security protocols described in [MS-SPSTWS].

1.6 Applicability Statement

This protocol is designed to retrieve user profile data for multiple users in one call. The number of users for which that data is retrieved is limited only by the configuration of the transport layer.

1.7 Versioning and Capability Negotiation

Supported Transports: This protocol can be implemented by using transports that support sending SOAP messages, as described in section 2.1.

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2 Messages

In the following sections, the schema definition might be less restrictive than the processing rules imposed by the protocol. The **WSDL** in this specification matches the WSDL that shipped with the product and provides a base description of the schema. The text that introduces the WSDL specifies additional restrictions that reflect protocol behavior. For example, the schema definition might allow for an element to be **empty**, **null**, or **not present** but the behavior of the protocol as specified restricts the same elements to being **non-empty**, **present**, and **not null**.

2.1 Transport

Protocol servers MUST support SOAP over HTTP, **HTTPS**, or **TCP**.

All protocol messages MUST be transported by using HTTP or TCP bindings at the transport level.

Protocol messages MUST be formatted as specified in either [SOAP1.1] section 4 or [SOAP1.2/1] section 5. Protocol server faults MUST be returned by using HTTP status codes, as specified in [RFC2616] section 10, or **SOAP faults**, as specified in [SOAP1.1] section 4.4 or [SOAP1.2/1] section 5.4.

If the HTTPS transport is used, a server certificate MUST be deployed.

This protocol MAY transmit an additional SOAP header, the **ServiceContext** header, as specified in [MS-SPSTWS].

This protocol does not define any means for activating a protocol server or protocol client. The protocol server MUST be configured and begin listening in an implementation-specific way. In addition, the protocol client MUST know the format and transport that is used by the server, for example, the SOAP format over an HTTP transport.

2.2 Common Message Syntax

This section contains common definitions that are used by this protocol. The syntax of the definitions uses the XML Schema syntax defined in [XMLSCHEMA1] and [XMLSCHEMA2], and Web Services Description Language, as defined in [WSDL].

2.2.1 Namespaces

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

2.2.2 Messages

None.

2.2.3 Elements

None.

2.2.4 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
ArrayOfString	Specifies an array of arbitrary strings.
InputFault	Specifies the details of an input error.
UnknownFault	Specifies the details of an unknown error.
UserProfileFault	Serves as an extension that can be used by all other error complex types.
ConfigurationFault	Specifies the details of a configuration error.

2.2.4.1 ConfigurationFault

Namespace: http://Microsoft/Office/Server/UserProfiles

The **ConfigurationFault** complex type describes errors that occur because of configuration failures either at the service endpoint (4) of this protocol or in transport layers. It is an extension of the **UserProfileFault** complex type.

```
<xs:complexType name="ConfigurationFault">
  <xs:complexContent mixed="false">
    <xs:extension base="tns:UserProfileFault">
      <xs:sequence/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

2.2.4.2 InputFault

Namespace: http://Microsoft/Office/Server/UserProfiles

The **InputFault** complex type describes errors that occur because of invalid **UserSearchCriteria** input from a protocol client. The **InputFault** complex type is an extension of the **UserProfileFault** complex type.

```
<xs:complexType name="InputFault">
  <xs:complexContent mixed="false">
    <xs:extension base="tns:UserProfileFault">
      <xs:sequence/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

2.2.4.3 UnknownFault

Namespace: http://Microsoft/Office/Server/UserProfiles

The **UnknownFault** complex type describes errors that occur for unknown reasons in the service endpoint (4) of this protocol. The **UnknownFault** complex type is an extension of the **UserProfileFault** complex type.

```
<xs:complexType name="UnknownFault">
  <xs:complexContent mixed="false">
    <xs:extension base="tns:UserProfileFault">
      <xs:sequence/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

2.2.4.4 UserProfileFault

Namespace: http://Microsoft/Office/Server/UserProfiles

The **UserProfileFault** complex type describes errors that occur because of errors within the service endpoint (4) of this protocol.

```
<xs:complexType name="UserProfileFault">
  <xs:sequence>
    <xs:element minOccurs="0" name="FailureDetail" nillable="true" type="xs:string"/>
    <xs:element minOccurs="0" name="FaultCode" type="tns:FaultCodes"/>
  </xs:sequence>
</xs:complexType>
```

FailureDetail: Specifies an error string. The protocol does not impose any restriction on the behavior of the protocol client with respect to this field.

FaultCode: Specifies a numerical indicator of the error that occurred. The protocol does not impose any restriction on the behavior of the protocol client with respect to this field.

2.2.5 Simple Types

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

Simple Type	Description
FaultCodes	An enumeration of possible errors.
guid	A globally unique identifier (GUID).

2.2.5.1 char

Namespace: http://schemas.microsoft.com/2003/10/Serialization/

The **char** simple type is a **Unicode** character that is stored in a 4-byte integer.

```
<xs:simpleType name="char">
  <xs:restriction base="xs:int"/>
</xs:simpleType>
```

2.2.5.2 duration

Namespace: http://schemas.microsoft.com/2003/10/Serialization/

The **duration** simple type is the length of time, as specified in [XMLSCHEMA].

```
<xs:simpleType name="duration">
  <xs:restriction base="xs:duration">
    <xs:pattern value="\-?P(\d*D)?(T(\d*H)?(\d*M)?(\d*(\.\d*)?S)?)?"/>
    <xs:minInclusive value="-P10675199DT2H48M5.4775808S"/>
    <xs:maxInclusive value="P10675199DT2H48M5.4775807S"/>
  </xs:restriction>
</xs:simpleType>
```

2.2.5.3 FaultCodes

Namespace: http://Microsoft/Office/Server/UserProfiles

The **FaultCodes** simple type is an enumeration of the possible errors that the service endpoint (4) of this protocol communicates with protocol clients. The protocol imposes no restriction on the behavior of the protocol client with respect to this type.

```
<xs:simpleType name="FaultCodes">
  <xs:restriction base="xs:string">
    <xs:enumeration value="UnknownError"/>
    <xs:enumeration value="InvalidInput"/>
    <xs:enumeration value="InvalidConfiguration"/>
  </xs:restriction>
</xs:simpleType>
```

The following table specifies the allowable values for FaultCodes:

Value	Meaning
UnknownError	The cause of the error is unknown to the service endpoint (4) of this protocol.
InvalidInput	The cause of the error is invalid **UserSearchCriteria** input from the protocol client.
InvalidConfiguration	The cause of the error is an invalid configuration of the service endpoint (4) of this protocol or the transport layers below it.

2.2.6 Attributes

None.

2.2.7 Groups

None.

2.2.8 Attribute Groups

None.

2.3 Directory Service Schema Elements

None.

3 Protocol Details

In the following sections, the schema definition might be less restrictive than the processing rules imposed by the protocol. The WSDL in this specification matches the WSDL that shipped with the product and provides a base description of the schema. The text that introduces the WSDL specifies additional restrictions that reflect protocol behavior. For example, the schema definition might allow for an element to be **empty**, **null**, or **not present** but the behavior of the protocol as specified restricts the same elements to being **non-empty**, **present**, and **not null**.

This protocol is a server-side protocol. The client side of this protocol is simply a pass-through. That is, no additional timers or other state is required on the client side of this protocol. Calls that are made by the upper protocol or application are passed directly to the transport, and the results returned by the transport are passed directly to the higher-layer protocol or application.

3.1 Server Details

This protocol is based on stateless interaction between the protocol client and protocol server. The protocol client MUST be authenticated with the credentials of a user who has access to the **User Profile Service** (referred to as 'UPA Standard Authentication' in the following figure). The protocol simply exposes the data that is currently stored in the User Profile Service that is deployed and running. There is no interdependency between the information exchanged between the protocol client and protocol server during one instance of their interaction using this protocol and the next instance. The following diagram illustrates the control flow for retrieving cached information about user profiles and returning it to the protocol client.

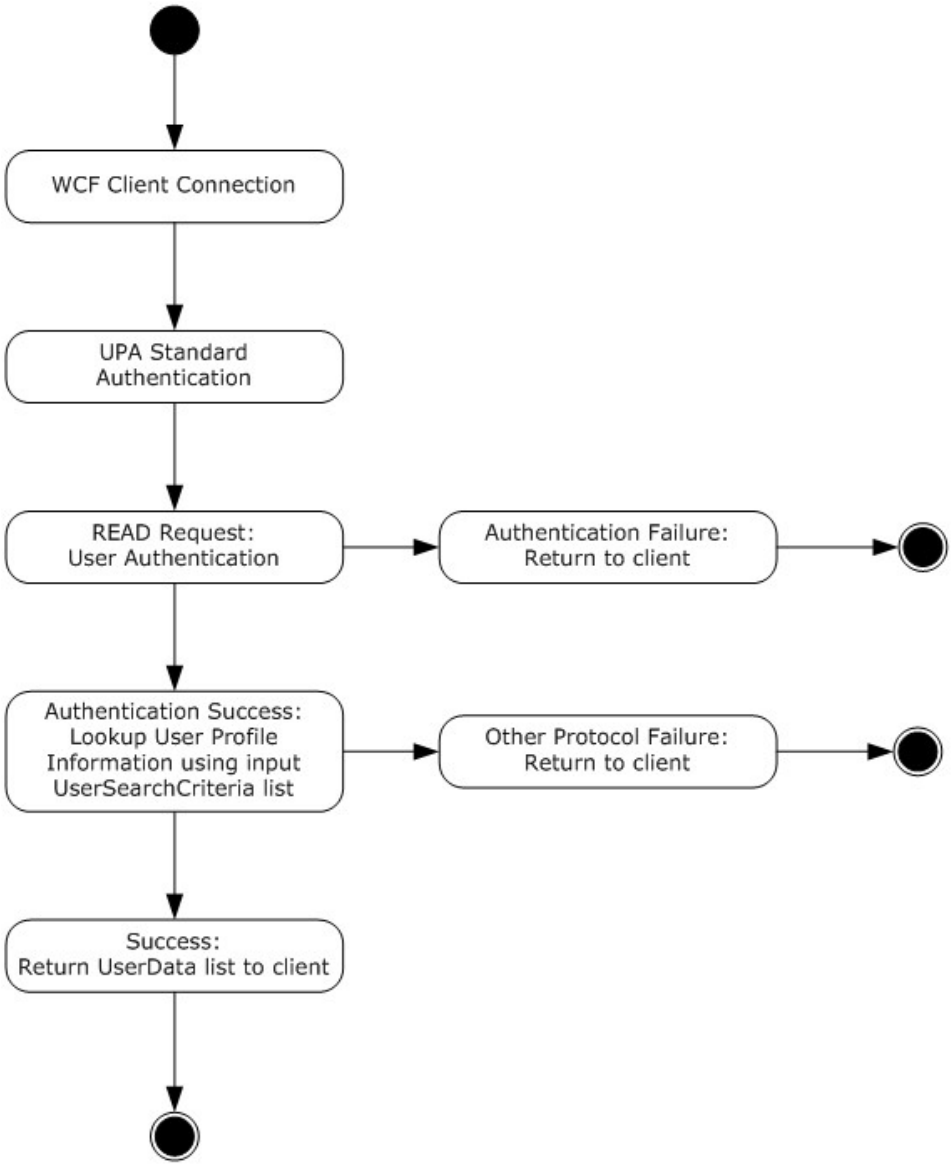

Figure 2: Control flow of the user profile caching service

When errors are returned to the protocol client, the service endpoint (4) of this protocol specifies complex types as described in sections 2.2.4.1 through 2.2.4.4 for each error.

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.

A key aspect of this protocol is that it allows protocol clients to gain quick and scalable access to a commonly used subset of User Profile Service properties and it does so through a WCF service endpoint (4) that caches these properties from the back-end database server, instead of requiring protocol clients to access the back-end database server directly. Although it is possible to use database connection strings that are exposed by the configuration state of the User Profile Application Service, as specified in [MS-UPWCFWS], and stored procedures, as specified in [MS-UPSCSP], it is recommended that protocol clients use the services that are offered by this protocol for commonly used user profile properties, such as e-mail, picture URL, user name, display name, and user profile record identifier. To access data that is not contained in this subset of properties, protocol clients can use stored procedures to gain access to the back-end database server directly. In addition, protocol clients can use stored procedures to gain access to the back-end database server directly when submitting changes to user profile information. Protocol clients cannot use this protocol to update user profile information; this protocol does not support write requests.

Because this protocol does not support write requests, the cache of user profile properties behind the service endpoint (4) of this protocol can potentially be out-of-date with the database layer before the cache is refreshed internally within the service. The following diagram captures these dependencies.

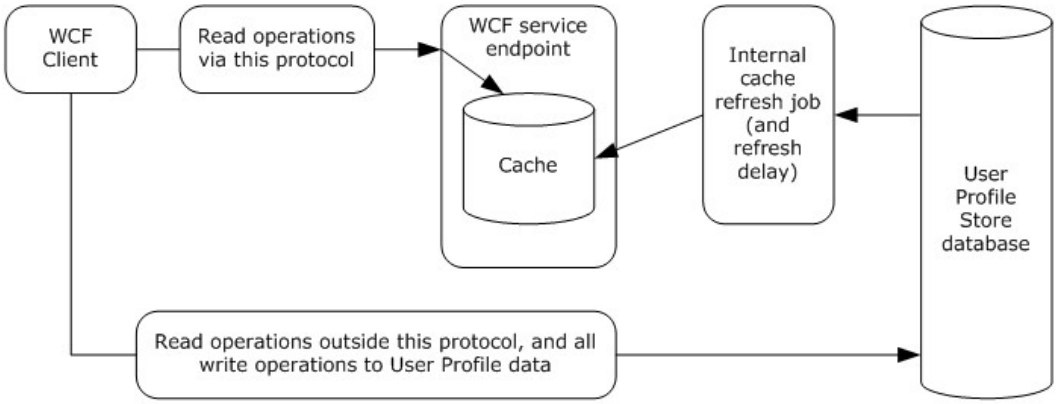

Figure 3: Dependencies on the cache

The preceding diagram illustrates that protocol clients that use this protocol MUST be able to tolerate a small delay, usually in the order of minutes, before a change that is made to the user profile data in the database is reflected in the cache. The cache refresh timer cannot be manipulated by using this protocol. It can be controlled only on the application server itself.

Initialization is not needed for the cache in the service endpoint (4) because the cache is populated on demand and transparently whenever the protocol client accesses data that is missing in the cache but present in the database. The protocol client interface is a very simple stateless request/response protocol.

3.1.2 Timers

None.

3.1.3 Initialization

None.

3.1.4 Message Processing Events and Sequencing Rules

This specification includes the following **WSDL operations**:

WSDL Operation	Description
GetUserData	The **GetUserData** operation retrieves the commonly used properties of a user profile in the **user profile store**.

3.1.4.1 GetUserData

The **GetUserData** operation retrieves the commonly used properties of a user profile in the user profile store.

```
<wsdl:operation name="GetUserData">
  <wsdl:input wsam:Action="http://Microsoft.Office.Server.UserProfiles/GetUserData"
message="tns:IProfileDBCacheService_GetUserData_InputMessage"/>
  <wsdl:output wsam:Action="http://tempuri.org/IProfileDBCacheService/GetUserDataResponse"
message="tns:IProfileDBCacheService_GetUserData_OutputMessage"/>
  <wsdl:fault
wsam:Action="http://tempuri.org/IProfileDBCacheService/GetUserDataInputFaultFault"
name="InputFaultFault"
message="tns:IProfileDBCacheService_GetUserData_InputFaultFault_FaultMessage"/>
  <wsdl:fault
wsam:Action="http://tempuri.org/IProfileDBCacheService/GetUserDataUnknownFaultFault"
name="UnknownFaultFault"
message="tns:IProfileDBCacheService_GetUserData_UnknownFaultFault_FaultMessage"/>
  <wsdl:fault
wsam:Action="http://tempuri.org/IProfileDBCacheService/GetUserDataUserProfileFaultFault"
name="UserProfileFaultFault"
message="tns:IProfileDBCacheService_GetUserData_UserProfileFaultFault_FaultMessage"/>
  <wsdl:fault
wsam:Action="http://tempuri.org/IProfileDBCacheService/GetUserDataConfigurationFaultFault"
name="ConfigurationFaultFault"
message="tns:IProfileDBCacheService_GetUserData_ConfigurationFaultFault_FaultMessage"/>
</wsdl:operation>
```

The protocol client sends an **IProfileDBCacheService_GetUserData_InputMessage** request message and the protocol server responds with an **IProfileDBCacheService_GetUserData_OutputMessage** response message. If the user profile store has user profile data that corresponds to the input **UserSearchCriteria** complex type, then this operation returns and packages that data for the protocol client as an output **UserData** complex type. The protocol client MAY send request to retrieve multiple user profiles in one call to this operation as specified in section 3.1.4.1.3.1. The operation returns one **UserData** complex type for each input that it receives, if the operation finds the user data. Otherwise, the operation MUST return **NULL** in place of **UserData** corresponding to the **UserSearchCriteria**. In case of any error, the operation MUST return an error, using one of the fault messages described in sections 2.2.4.1 through 2.2.4.4.

3.1.4.1.1 Messages

The following **WSDL message** definitions are specific to this operation.

3.1.4.1.1.1 IProfileDBCacheService_GetUserData_InputMessage

The request WSDL message for the **GetUserData** WSDL operation.

The **SOAP action** value is:

```
http://Microsoft.Office.Server.UserProfiles/GetUserData
```

The **SOAP body** contains the **GetUserData** element.

3.1.4.1.1.2 IProfileDBCacheService_GetUserData_OutputMessage

The response WSDL message for the **GetUserData** WSDL operation.

The SOAP body contains the **GetUserDataResponse** element.

3.1.4.1.2 Elements

The following XML Schema element definitions are specific to this operation.

3.1.4.1.2.1 GetUserData

The input data for the **GetUserData** WSDL operation.

```
<xs:element name="GetUserData">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" name="searchCriteria" nillable="true"
xmlns:q1="http://Microsoft/Office/Server/UserProfiles" type="q1:UserSearchCriteria"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

searchCriteria: Contains search criteria for the service endpoint (4) to use when searching for user profile data.

3.1.4.1.2.2 GetUserDataResponse

The result data for the **GetUserData** WSDL operation.

```
<xs:element name="GetUserDataResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" name="GetUserDataResult" nillable="true"
xmlns:q2="http://schemas.datacontract.org/2004/07/Microsoft.Office.Server.UserProfiles.Cache"
type="q2:StreamedDataOfArrayOfUserDatajHCugpoN"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

GetUserDataResult: Contains **StreamedDataOfArrayOfUserDatajHCugpoN** complex type. The collection of **UserData** complex types specified in **StreamedDataOfArrayOfUserDatajHCugpoN** complex type have a one-to-one relationship with input **searchCriteria** complex types. If the user

profile data for a specific **UserSearchCriteria** is not found, the corresponding **UserData** structure is nil.

3.1.4.1.3 Complex Types

The following XML Schema complex type definitions are specific to this operation.

3.1.4.1.3.1 UserSearchCriteria

Namespace: http://Microsoft/Office/Server/UserProfiles

The **UserSearchCriteria** complex type specifies the list of properties that the service endpoint (4) MUST use when searching for user profile data. In case of any error, the protocol server MUST return either an **InputFault**, **ConfigurationFault**, or **UnknownFault** complex type to the protocol client.

```
<xs:complexType name="UserSearchCriteria">
  <xs:sequence>
    <xs:element minOccurs="0" name="AllowAlternateAccountName" type="xs:boolean"/>
    <xs:element minOccurs="0" name="CorrelationID" type="ser:guid"/>
    <xs:element minOccurs="0" name="EmailCollection" nillable="true"
xmlns:q1="http://schemas.microsoft.com/2003/10/Serialization/Arrays"
type="q1:ArrayOfstring"/>
    <xs:element minOccurs="0" name="NTNameCollection" nillable="true"
xmlns:q2="http://schemas.microsoft.com/2003/10/Serialization/Arrays"
type="q2:ArrayOfstring"/>
    <xs:element minOccurs="0" name="PartitionID" type="ser:guid"/>
    <xs:element minOccurs="0" name="RecordIDCollection" nillable="true"
xmlns:q3="http://schemas.microsoft.com/2003/10/Serialization/Arrays" type="q3:ArrayOflong"/>
    <xs:element minOccurs="0" name="SIDCollection" nillable="true"
xmlns:q4="http://schemas.microsoft.com/2003/10/Serialization/Arrays"
type="q4:ArrayOfbase64Binary"/>
    <xs:element minOccurs="0" name="SearchColumn" nillable="true" type="xs:string"/>
    <xs:element minOccurs="0" name="UserIDCollection" nillable="true"
xmlns:q5="http://schemas.microsoft.com/2003/10/Serialization/Arrays" type="q5:ArrayOfguid"/>
  </xs:sequence>
</xs:complexType>
```

AllowAlternateAccountName: A single user can have multiple accounts in an organization. For example, a user can have an account "DOMAIN1\user1" for primary **authentication (1)** and another account "DOMAIN2\User1First.User1Last" for e-mail. When a user's user profile information is created, a distinct record for each user account is created in the user profile store, but the service ensures that all of the accounts have the same user profile data associated with them in the user profile store. The service also marks one of the accounts, for example "DOMAIN1\user1" as the master account, which is used primarily to identify the user. If this value is "true", it indicates that the **UserSearchCriteria** can be applied to any of the user's accounts. If this value is "false", it indicates that **UserSearchCriteria** MUST be applied only to the master account for the user.

CorrelationID: The optional **request identifier** for the current request.

EmailCollection: The list of e-mail addresses for users, based on which the search for their user profile data MUST be conducted by the service. A commonly used format is "user@domain.company".

NTNameCollection: The list of user names for users, based on which the search for their user profile data MUST be conducted by the service. A commonly used format is "DOMAIN\user".

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

RecordIDCollection: The list of 8-byte user profile record identifiers, based on which the search for the user profile data MUST be conducted by the service.

SIDCollection: The list of security identifiers (SIDs), based on which the search for the user profile data MUST be conducted by the service.

SearchColumn: Specifies which of the following search criteria fields to use when searching for user profile data:

- If the **SearchColumn** is the string "Email" it specifies **EmailCollection**.
- If the **SearchColumn** is the string "UserID" it specifies **UserIDCollection**.
- If the **SearchColumn** is the string "NTName" it specifies **NTNameCollection**.
- If the **SearchColumn** is the string "SID" it specifies **SIDCollection**.
- If the **SearchColumn** is the string "RecordID" it specifies **RecordIDCollection**.

The service never matches against more than one of these fields at a time. The service chooses exactly one string specified by **SearchColumn** to conduct a search. This value MUST NOT be null or empty.

UserIDCollection: The list of 16-byte **GUIDs** that uniquely identify a user, based on which the search for the user profile data MUST be conducted by the service.

3.1.4.1.3.2 ArrayOfstring

Namespace: http://schemas.microsoft.com/2003/10/Serialization/Arrays

ArrayOfstring is a sequence of zero or more strings.

```
<xs:complexType name="ArrayOfstring">
  <xs:sequence>
    <xs:element minOccurs="0" maxOccurs="unbounded" name="string" nillable="true"
type="xs:string"/>
  </xs:sequence>
</xs:complexType>
```

string: A single string value.

3.1.4.1.3.3 ArrayOflong

Namespace: http://schemas.microsoft.com/2003/10/Serialization/Arrays

ArrayOflong is a sequence of zero or more 8-byte, long integers.

```
<xs:complexType name="ArrayOflong">
  <xs:sequence>
    <xs:element minOccurs="0" maxOccurs="unbounded" name="long" type="xs:long"/>
  </xs:sequence>
</xs:complexType>
```

long: An 8-byte (64-bit) integer.

3.1.4.1.3.4 ArrayOfbase64Binary

Namespace: http://schemas.microsoft.com/2003/10/Serialization/Arrays

ArrayOfbase64Binary is a sequence of zero or more **base64Binary** types.

```
<xs:complexType name="ArrayOfbase64Binary">
  <xs:sequence>
    <xs:element minOccurs="0" maxOccurs="unbounded" name="base64Binary" nillable="true"
type="xs:base64Binary"/>
  </xs:sequence>
</xs:complexType>
```

base64Binary: This is a sequence of bytes represented in base 64 encoding.

3.1.4.1.3.5 ArrayOfguid

Namespace: http://schemas.microsoft.com/2003/10/Serialization/Arrays

ArrayOfguid is a sequence of zero or more 16-byte GUIDs.

```
<xs:complexType name="ArrayOfguid">
  <xs:sequence>
    <xs:element minOccurs="0" maxOccurs="unbounded" name="guid" type="ser:guid"/>
  </xs:sequence>
</xs:complexType>
```

guid: A 16-byte globally unique identifier (GUID).

3.1.4.1.3.6 StreamedDataOfArrayOfUserDatajHCugpoN

Namespace: http://schemas.datacontract.org/2004/07/Microsoft.Office.Server.UserProfiles.Cache

StreamedDataOfArrayOfUserDatajHCugpoN is a complex type that contains a stream with serialized contents of **ArrayOfUserData** complex type**.**

```
<xs:complexType name="StreamedDataOfArrayOfUserDatajHCugpoN">
  <xs:sequence>
    <xs:element minOccurs="0" name="DataStream" nillable="true"
xmlns:q1="http://schemas.datacontract.org/2004/07/System.IO" type="q1:MemoryStream"/>
  </xs:sequence>
</xs:complexType>
```

ArrayOfUserData: Specifies an **ArrayOfUserData** complex type.

DataStream: Specifies a **MemoryStream** complex type containing serialized contents of **ArrayOfUserData** complex type.

3.1.4.1.3.7 ArrayOfUserData

Namespace: http://Microsoft/Office/Server/UserProfiles

ArrayOfUserData is a sequence of zero or more **UserData** complex types.

```
<xs:complexType name="ArrayOfUserData">
  <xs:sequence>
    <xs:element minOccurs="0" maxOccurs="unbounded" name="UserData" nillable="true"
type="tns:UserData"/>
  </xs:sequence>
</xs:complexType>
```

UserData: Specifies a **UserData** complex type.

3.1.4.1.3.8 UserData

Namespace: http://Microsoft/Office/Server/UserProfiles

UserData is a complex type that contains the commonly used properties of the user profile data for a specific user in the user profile store.

```
<xs:complexType name="UserData">
  <xs:sequence>
    <xs:element minOccurs="0" name="Department" nillable="true" type="xs:string"/>
    <xs:element minOccurs="0" name="Email" nillable="true" type="xs:string"/>
    <xs:element minOccurs="0" name="LastUpdate" type="xs:dateTime"/>
    <xs:element minOccurs="0" name="MasterRecordID" type="xs:long"/>
    <xs:element minOccurs="0" name="NTName" nillable="true" type="xs:string"/>
    <xs:element minOccurs="1" name="PartitionID" type="ser:guid"/>
    <xs:element minOccurs="0" name="PictureUrl" nillable="true" type="xs:string"/>
    <xs:element minOccurs="0" name="PreferredName" nillable="true" type="xs:string"/>
    <xs:element minOccurs="0" name="ProfileSubtypeID" type="xs:int"/>
    <xs:element minOccurs="0" name="RecordID" type="xs:long"/>
    <xs:element minOccurs="0" name="SID" nillable="true" type="xs:base64Binary"/>
    <xs:element minOccurs="0" name="SipAddress" nillable="true" type="xs:string"/>
    <xs:element minOccurs="0" name="Title" nillable="true" type="xs:string"/>
    <xs:element minOccurs="0" name="UserID" type="ser:guid"/>
  </xs:sequence>
</xs:complexType>
```

Department: The department at work to which the user belongs.

Email: The work-related e-mail address of the user.

LastUpdate: The timestamp, in **UTC** format, when the user's user profile data was last modified.

MasterRecordID: The identifier of the primary record that stores the user profile data. If the **MasterRecordID** equals the **RecordID**, then this record MUST be the primary record of the user profile data of the user. The primary record is the main record for the user if the user has two user profiles.

NTName: The user name string that uniquely identifies the user within the authentication **domain** that is used to authenticate the user.

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

PictureUrl: The URL of a picture by which the user prefers to be recognized.

PreferredName: The display name of the user by which the user prefers to be addressed.

ProfileSubtypeID: An integer that indicates the type of profile. Valid values are "1" (user) and "2" (organization).

RecordID: An 8-byte, user profile record identifier.

SID: A security identifier (SID) that uniquely identifies the user within the authentication domain that is used to authenticate the user.

SipAddress: The **Session Initiation Protocol (SIP) address** of the user.

Title: The work title of the user.

UserID: A 16-byte GUID that uniquely identifies the user.

3.1.4.1.3.9 MemoryStream

Namespace: http://schemas.datacontract.org/2004/07/System.IO

MemoryStream is a complex type that specifies a **stream** whose backing store is memory.

```
<xs:complexType name="MemoryStream">
  <xs:complexContent mixed="false">
    <xs:extension base="tns:Stream">
      <xs:sequence>
        <xs:element name="_buffer" nillable="true" type="xs:base64Binary"/>
        <xs:element name="_capacity" type="xs:int"/>
        <xs:element name="_expandable" type="xs:boolean"/>
        <xs:element name="_exposable" type="xs:boolean"/>
        <xs:element name="_isOpen" type="xs:boolean"/>
        <xs:element name="_length" type="xs:int"/>
        <xs:element name="_origin" type="xs:int"/>
        <xs:element name="_position" type="xs:int"/>
        <xs:element name="_writable" type="xs:boolean"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

_buffer: The array of unsigned bytes from which the stream is created.

_capacity: Specifies the number of bytes allocated for this stream.

_expandable: Specifies if the stream is expandable.

_exposable: Specifies if the underlying byte buffer is exposed.

_isOpen: Specifies if the stream is open.

_length: Specifies the length of the stream in bytes.

_origin: Specifies the index into the underlying buffer at which the stream starts.

_position: Specifies the current position within the stream.

_writable: Specifies if the stream supports writing.

3.1.4.1.4 Simple Types

The following XML Schema simple type definitions are specific to this operation.

3.1.4.1.4.1 guid

Namespace: http://schemas.microsoft.com/2003/10/Serialization/

The **guid** simple type is a globally unique identifier that is 16 bytes long.

```
<xs:simpleType name="guid">
  <xs:restriction base="xs:string">
    <xs:pattern value="[\da-fA-F]{8}-[\da-fA-F]{4}-[\da-fA-F]{4}-[\da-fA-F]{4}-[\da-fA-
F]{12}"/>
  </xs:restriction>
</xs:simpleType>
```

3.1.4.1.5 Attributes

None.

3.1.4.1.6 Groups

None.

3.1.4.1.7 Attribute Groups

None.

3.1.5 Timer Events

None.

3.1.6 Other Local Events

None.

3.2 Client Details

The client side of this protocol is simply a pass-through. That is, no additional timers or other state is required on the client side of this protocol. Calls made by the higher-layer protocol or application are passed directly to the transport, and the results returned by the transport are passed directly to the higher-layer protocol or application.

3.2.1 Abstract Data Model

None.

3.2.2 Timers

None.

3.2.3 Initialization

None.

3.2.4 Message Processing Events and Sequencing Rules

None.

3.2.5 Timer Events

None.

3.2.6 Other Local Events

None.

4 Protocol Examples

This section provides specific example scenarios for retrieving a single user profile by **login name** and retrieving multiple user profiles by user profile record identifier.

Consider that the following two user profiles are part of the user profile store:

Record ID	User Name	NT Name	User ID
1	Agarwal, Nupur	contoso\nupuragarwal	4bccfd46-fb01-4c9b-988c-0730eaed529a
2	Cannon, Paul	contoso\paulcannon	b8110f57-8f8e-4c4a-9570-1c75828e18ef

The following identifiers are used in the examples in this section:

- PartitionID - 0C37852B-34D0-418E-91C6-2AC25AF4BE5B
- CorrelationID - a6b889ea-932a-4447-8d73-ce928cc912ad
- Security identifier (SID) - AQUAAAAAAAUVAAAAoGXPfnhLm1/nfIdw6iooAA==

4.1 Retrieving a User Profile by NT Name

To retrieve a user profile by domain user account, consider the following WSDL message constructed by the protocol client to call **GetUserData.**

```
  <s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
xmlns:a="http://www.w3.org/2005/08/addressing" xmlns:u="http://docs.oasis-
open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
    <s:Header>
      <a:Action
s:mustUnderstand="1">http://Microsoft.Office.Server.UserProfiles/GetUserData</a:Action>
      <a:MessageID>urn:uuid:3548f307-1764-4099-bbab-d421557cca8d</a:MessageID>
      <a:ReplyTo>
        <a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Address>
      </a:ReplyTo>
      <a:To
s:mustUnderstand="1">net.tcp://mymachine:32845/827f1817ac334b5da4476147b9ee5a8f/ProfileDBCach
eService.svc</a:To>
      <o:Security s:mustUnderstand="1" xmlns:o="http://docs.oasis-open.org/wss/2004/01/oasis-
200401-wss-wssecurity-secext-1.0.xsd">
        ...
      </o:Security>
    </s:Header>
    <s:Body>
      <GetUserData xmlns="http://tempuri.org/">
        <searchCriteria xmlns:b="http://Microsoft/Office/Server/UserProfiles"
xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
          <b:AllowAlternateAccountName>false</b:AllowAlternateAccountName>
          <b:CorrelationID>a6b889ea-932a-4447-8d73-ce928cc912ad</b:CorrelationID>
          <b:EmailCollection i:nil="true"
xmlns:c="http://schemas.microsoft.com/2003/10/Serialization/Arrays"></b:EmailCollection>
          <b:NTNameCollection
xmlns:c="http://schemas.microsoft.com/2003/10/Serialization/Arrays">
            <c:string>contoso\nupuragarwal</c:string>
          </b:NTNameCollection>
          <b:PartitionID>0c37852b-34d0-418e-91c6-2ac25af4be5b</b:PartitionID>
```

```
          <b:RecordIDCollection i:nil="true"
xmlns:c="http://schemas.microsoft.com/2003/10/Serialization/Arrays"></b:RecordIDCollection>
          <b:SIDCollection i:nil="true"
xmlns:c="http://schemas.microsoft.com/2003/10/Serialization/Arrays"></b:SIDCollection>
          <b:SearchColumn>NTName</b:SearchColumn>
          <b:UserIDCollection i:nil="true"
xmlns:c="http://schemas.microsoft.com/2003/10/Serialization/Arrays"></b:UserIDCollection>
        </searchCriteria>
      </GetUserData>
    </s:Body>
  </s:Envelope>
```

The protocol server returns the following WSDL message response with the requested data:

```
  <s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
xmlns:a="http://www.w3.org/2005/08/addressing" xmlns:u="http://docs.oasis-
open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
    <s:Header>
      <a:Action
s:mustUnderstand="1">http://tempuri.org/IProfileDBCacheService/GetUserDataResponse</a:Action>
      <a:RelatesTo>urn:uuid:3548f307-1764-4099-bbab-d421557cca8d</a:RelatesTo>
      <a:To s:mustUnderstand="1">http://www.w3.org/2005/08/addressing/anonymous</a:To>
      <o:Security s:mustUnderstand="1" xmlns:o="http://docs.oasis-open.org/wss/2004/01/oasis-
200401-wss-wssecurity-secext-1.0.xsd">
        ...
      </o:Security>
    </s:Header>
    <s:Body>
      <GetUserDataResponse xmlns="http://tempuri.org/">
        <GetUserDataResult xmlns:b="http://Microsoft/Office/Server/UserProfiles"
xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
          <d4p1:DataStream xmlns:d5p1="http://schemas.datacontract.org/2004/07/System.IO">
            <__identity i:nil="true"
xmlns="http://schemas.datacontract.org/2004/07/System"></__identity>
            <d5p1:_buffer></d5p1:_buffer>
            <d5p1:_capacity>672</d5p1:_capacity>
            <d5p1:_expandable>true</d5p1:_expandable>
            <d5p1:_exposable>true</d5p1:_exposable>
            <d5p1:_isOpen>true</d5p1:_isOpen>
            <d5p1:_length>672</d5p1:_length>
            <d5p1:_origin>0</d5p1:_origin>
            <d5p1:_position>0</d5p1:_position>
            <d5p1:_writable>true</d5p1:_writable>
          </d4p1:DataStream>
        </GetUserDataResult>
      </GetUserDataResponse>
    </s:Body>
  </s:Envelope>
```

4.2 Retrieving Multiple User Profiles by Record Identifier

To retrieve multiple user profiles by record identifier, consider the following WSDL message constructed by the protocol client to call **GetUserData**.

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
xmlns:a="http://www.w3.org/2005/08/addressing" xmlns:u="http://docs.oasis-
open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
```

```
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://Microsoft.Office.Server.UserProfiles/GetUserData</a:Action>
    <a:MessageID>urn:uuid:d0d7a499-931c-4e16-8407-6ea7f43b2a4a</a:MessageID>
    <a:ReplyTo>
      <a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Address>
    </a:ReplyTo>
    <a:To
s:mustUnderstand="1">net.tcp://mymachine:32845/827f1817ac334b5da4476147b9ee5a8f/ProfileDBCach
eService.svc</a:To>
    <o:Security s:mustUnderstand="1" xmlns:o="http://docs.oasis-open.org/wss/2004/01/oasis-
200401-wss-wssecurity-secext-1.0.xsd">
      ...
    </o:Security>
  </s:Header>
  <s:Body>
    <GetUserData xmlns="http://tempuri.org/">
      <searchCriteria xmlns:b="http://Microsoft/Office/Server/UserProfiles"
xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
        <b:AllowAlternateAccountName>true</b:AllowAlternateAccountName>
        <b:CorrelationID>a6b889ea-932a-4447-8d73-ce928cc912ad</b:CorrelationID>
        <b:EmailCollection i:nil="true"
xmlns:c="http://schemas.microsoft.com/2003/10/Serialization/Arrays"></b:EmailCollection>
        <b:NTNameCollection i:nil="true"
xmlns:c="http://schemas.microsoft.com/2003/10/Serialization/Arrays"></b:NTNameCollection>
        <b:PartitionID>0c37852b-34d0-418e-91c6-2ac25af4be5b</b:PartitionID>
        <b:RecordIDCollection xmlns:c="http://www.w3.org/2001/XMLSchema">
          <c:long>1</c:long>
          <c:long>2</c:long>
        </b:RecordIDCollection>
        <b:SIDCollection i:nil="true"
xmlns:c="http://schemas.microsoft.com/2003/10/Serialization/Arrays"></b:SIDCollection>
        <b:SearchColumn>RecordID</b:SearchColumn>
        <b:UserIDCollection i:nil="true"
xmlns:c="http://schemas.microsoft.com/2003/10/Serialization/Arrays"></b:UserIDCollection>
      </searchCriteria>
    </GetUserData>
  </s:Body>
</s:Envelope>
```

The protocol server responds with the following WSDL message which contains the requested data:

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
xmlns:a="http://www.w3.org/2005/08/addressing" xmlns:u="http://docs.oasis-
open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://tempuri.org/IProfileDBCacheService/GetUserDataResponse</a:Action>
    <a:RelatesTo>urn:uuid:d0d7a499-931c-4e16-8407-6ea7f43b2a4a</a:RelatesTo>
    <a:To s:mustUnderstand="1">http://www.w3.org/2005/08/addressing/anonymous</a:To>
    <o:Security s:mustUnderstand="1" xmlns:o="http://docs.oasis-open.org/wss/2004/01/oasis-
200401-wss-wssecurity-secext-1.0.xsd">
      ...
    </o:Security>
  </s:Header>
  <s:Body>
    <GetUserDataResponse xmlns="http://tempuri.org/">
      <GetUserDataResult xmlns:b="http://Microsoft/Office/Server/UserProfiles"
xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
```

```
        <d4p1:DataStream xmlns:d5p1="http://schemas.datacontract.org/2004/07/System.IO">
          <__identity i:nil="true"
xmlns="http://schemas.datacontract.org/2004/07/System"></__identity>
          <d5p1:_buffer></d5p1:_buffer>
          <d5p1:_capacity>2054</d5p1:_capacity>
          <d5p1:_expandable>true</d5p1:_expandable>
          <d5p1:_exposable>true</d5p1:_exposable>
          <d5p1:_isOpen>true</d5p1:_isOpen>
          <d5p1:_length>1205</d5p1:_length>
          <d5p1:_origin>0</d5p1:_origin>
          <d5p1:_position>0</d5p1:_position>
          <d5p1:_writable>true</d5p1:_writable>
        </d4p1:DataStream>
      </GetUserDataResult>
    </GetUserDataResponse>
  </s:Body>
</s:Envelope>
```

5 Security

5.1 Security Considerations for Implementers

There are no security considerations that are specific to this protocol. General security considerations pertaining to [RFC2822] apply.

This protocol does not introduce any additional security considerations beyond those that apply to its underlying protocols.

5.2 Index of Security Parameters

None.

6 Appendix A: Full WSDL

For ease of implementation, the following full WSDL is provided:

```
<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions xmlns:tns="http://tempuri.org/"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:wsam="http://www.w3.org/2007/05/addressing/metadata"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" name="ProfileDBCacheService"
targetNamespace="http://tempuri.org/" xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
  <wsdl:types>
    <xs:schema elementFormDefault="qualified" targetNamespace="http://tempuri.org/"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
      <xs:import namespace="http://Microsoft/Office/Server/UserProfiles" />
      <xs:import
namespace="http://schemas.datacontract.org/2004/07/Microsoft.Office.Server.UserProfiles.Cache
" />
      <xs:element name="GetUserData">
        <xs:complexType>
          <xs:sequence>
            <xs:element xmlns:q1="http://Microsoft/Office/Server/UserProfiles" minOccurs="0"
name="searchCriteria" nillable="true" type="q1:UserSearchCriteria" />
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="GetUserDataResponse">
        <xs:complexType>
          <xs:sequence>
            <xs:element
xmlns:q2="http://schemas.datacontract.org/2004/07/Microsoft.Office.Server.UserProfiles.Cache"
minOccurs="0" name="GetUserDataResult" nillable="true"
type="q2:StreamedDataOfArrayOfUserDatajHCugpoN" />
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:schema>
    <xs:schema xmlns:tns="http://Microsoft/Office/Server/UserProfiles"
xmlns:ser="http://schemas.microsoft.com/2003/10/Serialization/"
elementFormDefault="qualified" targetNamespace="http://Microsoft/Office/Server/UserProfiles"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
      <xs:import namespace="http://schemas.microsoft.com/2003/10/Serialization/" />
      <xs:import namespace="http://schemas.microsoft.com/2003/10/Serialization/Arrays" />
      <xs:complexType name="UserSearchCriteria">
        <xs:sequence>
          <xs:element minOccurs="0" name="AllowAlternateAccountName" type="xs:boolean" />
          <xs:element minOccurs="0" name="CorrelationID" type="ser:guid" />
          <xs:element xmlns:q1="http://schemas.microsoft.com/2003/10/Serialization/Arrays"
minOccurs="0" name="EmailCollection" nillable="true" type="q1:ArrayOfstring" />
          <xs:element xmlns:q2="http://schemas.microsoft.com/2003/10/Serialization/Arrays"
minOccurs="0" name="NTNameCollection" nillable="true" type="q2:ArrayOfstring" />
          <xs:element minOccurs="1" name="PartitionID" type="ser:guid" />
          <xs:element xmlns:q3="http://schemas.microsoft.com/2003/10/Serialization/Arrays"
minOccurs="0" name="RecordIDCollection" nillable="true" type="q3:ArrayOflong" />
          <xs:element xmlns:q4="http://schemas.microsoft.com/2003/10/Serialization/Arrays"
minOccurs="0" name="SIDCollection" nillable="true" type="q4:ArrayOfbase64Binary" />
          <xs:element minOccurs="0" name="SearchColumn" nillable="true" type="xs:string" />
          <xs:element xmlns:q5="http://schemas.microsoft.com/2003/10/Serialization/Arrays"
minOccurs="0" name="UserIDCollection" nillable="true" type="q5:ArrayOfguid" />
        </xs:sequence>
      </xs:complexType>
```

```
      <xs:element name="UserSearchCriteria" nillable="true" type="tns:UserSearchCriteria" />
      <xs:complexType name="UserProfileFault">
        <xs:sequence>
          <xs:element minOccurs="0" name="FailureDetail" nillable="true" type="xs:string" />
          <xs:element minOccurs="0" name="FaultCode" type="tns:FaultCodes" />
        </xs:sequence>
      </xs:complexType>
      <xs:element name="UserProfileFault" nillable="true" type="tns:UserProfileFault" />
      <xs:simpleType name="FaultCodes">
        <xs:restriction base="xs:string">
          <xs:enumeration value="UnknownError" />
          <xs:enumeration value="InvalidInput" />
          <xs:enumeration value="InvalidConfiguration" />
        </xs:restriction>
      </xs:simpleType>
      <xs:element name="FaultCodes" nillable="true" type="tns:FaultCodes" />
      <xs:complexType name="InputFault">
        <xs:complexContent mixed="false">
          <xs:extension base="tns:UserProfileFault">
            <xs:sequence />
          </xs:extension>
        </xs:complexContent>
      </xs:complexType>
      <xs:element name="InputFault" nillable="true" type="tns:InputFault" />
      <xs:complexType name="ConfigurationFault">
        <xs:complexContent mixed="false">
          <xs:extension base="tns:UserProfileFault">
            <xs:sequence />
          </xs:extension>
        </xs:complexContent>
      </xs:complexType>
      <xs:element name="ConfigurationFault" nillable="true" type="tns:ConfigurationFault" />
      <xs:complexType name="UnknownFault">
        <xs:complexContent mixed="false">
          <xs:extension base="tns:UserProfileFault">
            <xs:sequence />
          </xs:extension>
        </xs:complexContent>
      </xs:complexType>
      <xs:element name="UnknownFault" nillable="true" type="tns:UnknownFault" />
      <xs:complexType name="ArrayOfUserData">
        <xs:sequence>
          <xs:element minOccurs="0" maxOccurs="unbounded" name="UserData" nillable="true"
type="tns:UserData" />
        </xs:sequence>
      </xs:complexType>
      <xs:element name="ArrayOfUserData" nillable="true" type="tns:ArrayOfUserData" />
      <xs:complexType name="UserData">
        <xs:sequence>
          <xs:element minOccurs="0" name="Department" nillable="true" type="xs:string" />
          <xs:element minOccurs="0" name="Email" nillable="true" type="xs:string" />
          <xs:element minOccurs="0" name="LastUpdate" type="xs:dateTime" />
          <xs:element minOccurs="0" name="MasterRecordID" type="xs:long" />
          <xs:element minOccurs="0" name="NTName" nillable="true" type="xs:string" />
          <xs:element minOccurs="1" name="PartitionID" type="ser:guid" />
          <xs:element minOccurs="0" name="PictureUrl" nillable="true" type="xs:string" />
          <xs:element minOccurs="0" name="PreferredName" nillable="true" type="xs:string" />
          <xs:element minOccurs="0" name="ProfileSubtypeID" type="xs:int" />
          <xs:element minOccurs="0" name="RecordID" type="xs:long" />
```

```
          <xs:element minOccurs="0" name="SID" nillable="true" type="xs:base64Binary" />
          <xs:element minOccurs="0" name="SipAddress" nillable="true" type="xs:string" />
          <xs:element minOccurs="0" name="Title" nillable="true" type="xs:string" />
          <xs:element minOccurs="0" name="UserID" type="ser:guid" />
        </xs:sequence>
      </xs:complexType>
      <xs:element name="UserData" nillable="true" type="tns:UserData" />
    </xs:schema>
    <xs:schema xmlns:tns="http://schemas.microsoft.com/2003/10/Serialization/"
attributeFormDefault="qualified" elementFormDefault="qualified"
targetNamespace="http://schemas.microsoft.com/2003/10/Serialization/"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
      <xs:element name="anyType" nillable="true" type="xs:anyType" />
      <xs:element name="anyURI" nillable="true" type="xs:anyURI" />
      <xs:element name="base64Binary" nillable="true" type="xs:base64Binary" />
      <xs:element name="boolean" nillable="true" type="xs:boolean" />
      <xs:element name="byte" nillable="true" type="xs:byte" />
      <xs:element name="dateTime" nillable="true" type="xs:dateTime" />
      <xs:element name="decimal" nillable="true" type="xs:decimal" />
      <xs:element name="double" nillable="true" type="xs:double" />
      <xs:element name="float" nillable="true" type="xs:float" />
      <xs:element name="int" nillable="true" type="xs:int" />
      <xs:element name="long" nillable="true" type="xs:long" />
      <xs:element name="QName" nillable="true" type="xs:QName" />
      <xs:element name="short" nillable="true" type="xs:short" />
      <xs:element name="string" nillable="true" type="xs:string" />
      <xs:element name="unsignedByte" nillable="true" type="xs:unsignedByte" />
      <xs:element name="unsignedInt" nillable="true" type="xs:unsignedInt" />
      <xs:element name="unsignedLong" nillable="true" type="xs:unsignedLong" />
      <xs:element name="unsignedShort" nillable="true" type="xs:unsignedShort" />
      <xs:element name="char" nillable="true" type="tns:char" />
      <xs:simpleType name="char">
        <xs:restriction base="xs:int" />
      </xs:simpleType>
      <xs:element name="duration" nillable="true" type="tns:duration" />
      <xs:simpleType name="duration">
        <xs:restriction base="xs:duration">
          <xs:pattern value="\-?P(\d*D)?(T(\d*H)?(\d*M)?(\d*(\.\d*)?S)?)?" />
          <xs:minInclusive value="-P10675199DT2H48M5.4775808S" />
          <xs:maxInclusive value="P10675199DT2H48M5.4775807S" />
        </xs:restriction>
      </xs:simpleType>
      <xs:element name="guid" nillable="true" type="tns:guid" />
      <xs:simpleType name="guid">
        <xs:restriction base="xs:string">
          <xs:pattern value="[\da-fA-F]{8}-[\da-fA-F]{4}-[\da-fA-F]{4}-[\da-fA-F]{4}-[\da-fA-
F]{12}" />
        </xs:restriction>
      </xs:simpleType>
      <xs:attribute name="FactoryType" type="xs:QName" />
      <xs:attribute name="Id" type="xs:ID" />
      <xs:attribute name="Ref" type="xs:IDREF" />
    </xs:schema>
    <xs:schema xmlns:tns="http://schemas.microsoft.com/2003/10/Serialization/Arrays"
xmlns:ser="http://schemas.microsoft.com/2003/10/Serialization/"
elementFormDefault="qualified"
targetNamespace="http://schemas.microsoft.com/2003/10/Serialization/Arrays"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
      <xs:import namespace="http://schemas.microsoft.com/2003/10/Serialization/" />
      <xs:complexType name="ArrayOfstring">
```

```
        <xs:sequence>
          <xs:element minOccurs="0" maxOccurs="unbounded" name="string" nillable="true"
type="xs:string" />
        </xs:sequence>
      </xs:complexType>
      <xs:element name="ArrayOfstring" nillable="true" type="tns:ArrayOfstring" />
      <xs:complexType name="ArrayOflong">
        <xs:sequence>
          <xs:element minOccurs="0" maxOccurs="unbounded" name="long" type="xs:long" />
        </xs:sequence>
      </xs:complexType>
      <xs:element name="ArrayOflong" nillable="true" type="tns:ArrayOflong" />
      <xs:complexType name="ArrayOfbase64Binary">
        <xs:sequence>
          <xs:element minOccurs="0" maxOccurs="unbounded" name="base64Binary" nillable="true"
type="xs:base64Binary" />
        </xs:sequence>
      </xs:complexType>
      <xs:element name="ArrayOfbase64Binary" nillable="true" type="tns:ArrayOfbase64Binary"
/>
      <xs:complexType name="ArrayOfguid">
        <xs:sequence>
          <xs:element minOccurs="0" maxOccurs="unbounded" name="guid" type="ser:guid" />
        </xs:sequence>
      </xs:complexType>
      <xs:element name="ArrayOfguid" nillable="true" type="tns:ArrayOfguid" />
    </xs:schema>
    <xs:schema
xmlns:tns="http://schemas.datacontract.org/2004/07/Microsoft.Office.Server.UserProfiles.Cache
" elementFormDefault="qualified"
targetNamespace="http://schemas.datacontract.org/2004/07/Microsoft.Office.Server.UserProfiles
.Cache" xmlns:xs="http://www.w3.org/2001/XMLSchema">
      <xs:import namespace="http://schemas.datacontract.org/2004/07/System.IO" />
      <xs:complexType name="StreamedDataOfArrayOfUserDatajHCugpoN">
        <xs:sequence>
          <xs:element xmlns:q1="http://schemas.datacontract.org/2004/07/System.IO"
minOccurs="0" name="DataStream" nillable="true" type="q1:MemoryStream" />
        </xs:sequence>
      </xs:complexType>
      <xs:element name="StreamedDataOfArrayOfUserDatajHCugpoN" nillable="true"
type="tns:StreamedDataOfArrayOfUserDatajHCugpoN" />
    </xs:schema>
    <xs:schema xmlns:tns="http://schemas.datacontract.org/2004/07/System.IO"
elementFormDefault="qualified"
targetNamespace="http://schemas.datacontract.org/2004/07/System.IO"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
      <xs:import namespace="http://schemas.datacontract.org/2004/07/System" />
      <xs:complexType name="MemoryStream">
        <xs:complexContent mixed="false">
          <xs:extension base="tns:Stream">
            <xs:sequence>
              <xs:element name="_buffer" nillable="true" type="xs:base64Binary" />
              <xs:element name="_capacity" type="xs:int" />
              <xs:element name="_expandable" type="xs:boolean" />
              <xs:element name="_exposable" type="xs:boolean" />
              <xs:element name="_isOpen" type="xs:boolean" />
              <xs:element name="_length" type="xs:int" />
              <xs:element name="_origin" type="xs:int" />
              <xs:element name="_position" type="xs:int" />
              <xs:element name="_writable" type="xs:boolean" />
```

```
            </xs:sequence>
          </xs:extension>
        </xs:complexContent>
      </xs:complexType>
      <xs:element name="MemoryStream" nillable="true" type="tns:MemoryStream" />
      <xs:complexType name="Stream">
        <xs:complexContent mixed="false">
          <xs:extension xmlns:q1="http://schemas.datacontract.org/2004/07/System"
base="q1:MarshalByRefObject">
            <xs:sequence />
          </xs:extension>
        </xs:complexContent>
      </xs:complexType>
      <xs:element name="Stream" nillable="true" type="tns:Stream" />
    </xs:schema>
    <xs:schema xmlns:tns="http://schemas.datacontract.org/2004/07/System"
elementFormDefault="qualified"
targetNamespace="http://schemas.datacontract.org/2004/07/System"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
      <xs:complexType name="MarshalByRefObject">
        <xs:sequence>
          <xs:element name="__identity" nillable="true" type="xs:anyType" />
        </xs:sequence>
      </xs:complexType>
      <xs:element name="MarshalByRefObject" nillable="true" type="tns:MarshalByRefObject" />
    </xs:schema>
  </wsdl:types>
  <wsdl:message name="IProfileDBCacheService_GetUserData_InputMessage">
    <wsdl:part name="parameters" element="tns:GetUserData" />
  </wsdl:message>
  <wsdl:message name="IProfileDBCacheService_GetUserData_OutputMessage">
    <wsdl:part name="parameters" element="tns:GetUserDataResponse" />
  </wsdl:message>
  <wsdl:message name="IProfileDBCacheService_GetUserData_UserProfileFaultFault_FaultMessage">
    <wsdl:part xmlns:q1="http://Microsoft/Office/Server/UserProfiles" name="detail"
element="q1:UserProfileFault" />
  </wsdl:message>
  <wsdl:message name="IProfileDBCacheService_GetUserData_InputFaultFault_FaultMessage">
    <wsdl:part xmlns:q2="http://Microsoft/Office/Server/UserProfiles" name="detail"
element="q2:InputFault" />
  </wsdl:message>
  <wsdl:message
name="IProfileDBCacheService_GetUserData_ConfigurationFaultFault_FaultMessage">
    <wsdl:part xmlns:q3="http://Microsoft/Office/Server/UserProfiles" name="detail"
element="q3:ConfigurationFault" />
  </wsdl:message>
  <wsdl:message name="IProfileDBCacheService_GetUserData_UnknownFaultFault_FaultMessage">
    <wsdl:part xmlns:q4="http://Microsoft/Office/Server/UserProfiles" name="detail"
element="q4:UnknownFault" />
  </wsdl:message>
  <wsdl:portType name="IProfileDBCacheService">
    <wsdl:operation name="GetUserData">
      <wsdl:input wsam:Action="http://Microsoft.Office.Server.UserProfiles/GetUserData"
message="tns:IProfileDBCacheService_GetUserData_InputMessage" />
      <wsdl:output
wsam:Action="http://tempuri.org/IProfileDBCacheService/GetUserDataResponse"
message="tns:IProfileDBCacheService_GetUserData_OutputMessage" />
      <wsdl:fault
wsam:Action="http://tempuri.org/IProfileDBCacheService/GetUserDataUserProfileFaultFault"
```

```
name="UserProfileFaultFault"
message="tns:IProfileDBCacheService_GetUserData_UserProfileFaultFault_FaultMessage" />
      <wsdl:fault
wsam:Action="http://tempuri.org/IProfileDBCacheService/GetUserDataInputFaultFault"
name="InputFaultFault"
message="tns:IProfileDBCacheService_GetUserData_InputFaultFault_FaultMessage" />
      <wsdl:fault
wsam:Action="http://tempuri.org/IProfileDBCacheService/GetUserDataConfigurationFaultFault"
name="ConfigurationFaultFault"
message="tns:IProfileDBCacheService_GetUserData_ConfigurationFaultFault_FaultMessage" />
      <wsdl:fault
wsam:Action="http://tempuri.org/IProfileDBCacheService/GetUserDataUnknownFaultFault"
name="UnknownFaultFault"
message="tns:IProfileDBCacheService_GetUserData_UnknownFaultFault_FaultMessage" />
    </wsdl:operation>
  </wsdl:portType>
  <wsdl:binding name="CustomBinding_IProfileDBCacheService"
type="tns:IProfileDBCacheService">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http" />
    <wsdl:operation name="GetUserData">
      <soap:operation soapAction="http://Microsoft.Office.Server.UserProfiles/GetUserData"
style="document" />
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
      <wsdl:fault name="UserProfileFaultFault">
        <soap:fault use="literal" name="UserProfileFaultFault" namespace="" />
      </wsdl:fault>
      <wsdl:fault name="InputFaultFault">
        <soap:fault use="literal" name="InputFaultFault" namespace="" />
      </wsdl:fault>
      <wsdl:fault name="ConfigurationFaultFault">
        <soap:fault use="literal" name="ConfigurationFaultFault" namespace="" />
      </wsdl:fault>
      <wsdl:fault name="UnknownFaultFault">
        <soap:fault use="literal" name="UnknownFaultFault" namespace="" />
      </wsdl:fault>
    </wsdl:operation>
  </wsdl:binding>
</wsdl:definitions>
```

7 Appendix B: 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.

8 Change Tracking

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

9 Index