



SuccessFactors

People Performance

PartnerService User Guide

Importing Users

Revision			
Date	Author	Version	Description
4/20/2009	Paul Littlefield	1.0	Initial Draft – Login, Logout, User Import
12/28/2009	Paul Littlefield	1.1	Updated error handling section

Table of Contents

Table of Contents	2
Introduction.....	3
EndPoints	3
WSDL	3
Development Process.....	3
Login	4
SFCredential	4
LoginResult (Extended from SFObject)	4
Managing the Session ID	5
Example SOAP Request for Login	5
Example SOAP Response for Login.....	5
Logout	5
Example SOAP Request for Logout	6
Example SOAP Response for Logout.....	6
Importing Users	6
UserObject (Extended from SFObject).....	6
SFObject	7
SFAttribute (Extended from SFObject)	7
SFAttributes for UserObject	7
PutResult (Extended from SFObject – permissioned access).....	9
Handling Errors.....	9
SFWebServiceError (Extended from SFObject)	9
Error Descriptions and Error Numbers	9
Other PartnerService API Error Codes	12
Example SOAP Request for Put UserObject	13
Example SOAP Response for Put UserObject – Successful	14
Example SOAP Response for Put UserObject with Errors	14

Introduction

The SuccessFactors PartnerService API is a SOAP Web Service that can be used to programmatically exchange data with a SuccessFactors hosted application instance. It is implemented in SOAP 1.1 with an RPC/encoded WSDL.

This document is a user guide focused on importing user into the SuccessFactors system. For a broader reference guide on the API please see the document titled “SuccessFactors PartnerService API Reference Guide v1.6”.

EndPoints

Your endpoint will depend on where your SuccessFactors instance is located. It can be one of three data centers. Please contact your SuccessFactors representative if you are unsure of which data center to use. The endpoints are below.

Data Center Location	Endpoint URL
Europe	https://soap.successfactors.eu/axis/services/PartnerService
USA, New Jersey	https://soap.successfactors.com/axis/services/PartnerService
USA, Arizona	https://soap4.successfactors.com/axis/services/PartnerService

WSDL

The WSDL can be accessed by appending “?wsdl” to the endpoint.

Data Center Location	WSDL
Europe	https://soap.successfactors.eu/axis/services/PartnerService?wsdl
USA, New Jersey	https://soap.successfactors.com/axis/services/PartnerService?wsdl
USA, Arizona	https://soap4.successfactors.com/axis/services/PartnerService?wsdl

Development Process

In order to access the Partner API it must be enabled for your company instance. Contact your SuccessFactors Professional Services or Customer Success representative if you want the Partner API enabled.

Once the API is enabled you can access your instance through the API using any valid user. The API interaction is session based, meaning that the first step is to login to establish an API session. Then you may invoke put and get calls. When you are finished, you may logout, or let the session expire automatically after 30 minutes of activity (this is known as a session timeout).

It is your choice whether you develop and test against your live production instance or a separate test instance. Test instances typically incur additional costs. Ask your SuccessFactors representative if you are interested in a test instance.

Login

Login is accomplished from the Login method call. Users must be sure they are contacting the correct web services endpoint for their SuccessFactors instance. Contact your SuccessFactors support representative if you are unsure.

Below is a listing of the various endpoints that exist today.

Data Center Location	Endpoint URL
Europe	https://soap.successfactors.eu/axis/services/PartnerService
USA, New Jersey	https://soap.successfactors.com/axis/services/PartnerService
USA, Arizona	https://soap4.successfactors.com/axis/services/PartnerService

Once you have the correct endpoint, login is accomplished with the login method call. This has the signature listed below

```
LoginResult login( SFCredential )
```

The SFCredential and LoginResult objects are documented below. In the SFCredential object, only three fields are actually used: CompanyID, Username and Password.

SFCredential

This must be passed in the [login](#) method.

Name	Type	Description
PartnerID	String	
CompanyID	String	Identifies the SuccessFactors client instance.
Username	String	Username (must be a valid user in the specified client instance).
Password	String	Password associated with the Username.
clientIP	String	
Expire	String	
Timezone	String	
tkloginKey	String	
callerHash	String	

LoginResult (Extended from SFObject)

This object is returned from a successful [login](#) call.

Name	Type	Description
URL	String	Not used
SessionId	String	The session identifier assigned by the server.

Managing the Session ID

The LoginResult will contain the SessionId for the login. Rather than manage the sessionId yourself, most users will typically enable cookies in the HTTP session, and let the HTTP protocol manage the session attribute for you. Then the login request (SOAP over HTTP) will set the session Id attribute as a cookie.

Once a login is established, it will be valid until either a logout() call is made, or the session timeout is reached (30 minutes) due to inactivity on the session.

Example SOAP Request for Login

```
<soapenv:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:partnerservice="http://partnerService.successfactors.com">
  <soapenv:Header/>
  <soapenv:Body>
    <partnerservice:login>
      <credential>
        <companyId xsi:type="xsd:string">ACE123</companyId>
        <username xsi:type="xsd:string">cgrant</username>
        <password xsi:type="xsd:string">password-string</password>
      </credential>
    </partnerservice:login>
  </soapenv:Body>
</soapenv:Envelope>
```

Example SOAP Response for Login

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance">
  <soapenv:Body>
    <ns1:loginResponse soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://server.partnerService.axis.sf4.sf.com">
      <loginReturn href="#id0"/>
    </ns1:loginResponse>
    <multiRef id="id0" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" xsi:type="ns2:LoginResult"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/" xmlns:ns2="urn:PartnerService">
      <lastModified xsi:type="soapenc:string" xsi:nil="true"/>
      <locale xsi:type="soapenc:string" xsi:nil="true"/>
      <objectId xsi:type="soapenc:string" xsi:nil="true"/>
      <sessionId xsi:type="soapenc:string">3389B6BB58EB7112DC80DD7D7FF0A60E </sessionId>
      <sfAttributes xsi:type="soapenc:Array" xsi:nil="true"/>
      <url xsi:type="soapenc:string"/>
    </multiRef>
  </soapenv:Body>
</soapenv:Envelope>
```

Logout

Logout is accomplished with the logout() call. There are no parameters to this call, and no values are returned.

Example SOAP Request for Logout

```
<soapenv:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:ns1="http://server.axis.sfv4.sf.com">
  <soapenv:Header/>
  <soapenv:Body>
    <ns1:logout soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding"/>
  </soapenv:Body>
</soapenv:Envelope>
```

Example SOAP Response for Logout

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance">
  <soapenv:Body>
    <ns1:logoutResponse soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://server.axis.sfv4.sf.com"/>
  </soapenv:Body>
</soapenv:Envelope>
```

Importing Users

Importing users with the PartnerService API is accomplished through the “put” method by passing in an array of “UserObjects”. The call signature for put is:

```
PutResult put( String, SFObject[])
```

This API uses a polymorphic approach, so you can call the put method with an array of objects that subclass the generic SFObject class. UserObject is the subtype to use to import users.

The “UserObject” is defined in the table below. It has only two strongly typed fields (Username and UserID).

UserObject (Extended from SFObject)		
System user record. User fields not explicitly defined here are added as SFAttribute objects.		
Name	Type	Description
Username	String	Unique username
UserID	String	Unique userID (could be same as username)

In addition to the attributes listed above, SFObject inherits from the SFObject class (it extends the SFObject class) and therefore has an array of SFAttributes objects. An SFAttribute is a generic container object which are basically name/value pairs of information. This array of SFAttributes is utilized to import the rest of the fields desired

for importing data in a UserObject. A separate table below lists the valid values allowed in SFAttributes when importing a UserObject.

The SFObject and SFAttribute objects are documented in the tables below.

SFObject		
Base SuccessFactors object		
Name	Type	Description
ObjectID	String	Unique identifier of the object.
Locale	String	Describes the locale of the user session.
LastModified	String	Last modified date, in format “yyyy-mm-ddThh:mi:ss”
SFAttributes	SFAttribute[]	Optional list of attributes to handle custom fields

SFAttribute (Extended from SFObject)		
A name/value pair used as custom fields for each object.		
Name	Type	Description
Name	String	Describes the attribute. Examples: “Planned”, “Registered”, “Start”, “Completed”
Value	String	Value corresponding to the attribute
Type	String	Data type definition of the attribute. Examples “String”, “Date”, “Number”

SFAttributes for UserObject

The following is a list of attributes that may be set for a UserObject when importing. For all of these attributes, set the Type to “String” in the SFAttribute object. For example,

```
<SFAttribute >
  <Name>FIRSTNAME</Name>
  <Type>String</Type>
  <Value>John</Value>
</SFAttribute>
```

Name	Example	Description
STATUS	active	Valid values (case sensitive): active, inactive, transfer, active_external, inactive_external
FIRSTNAME	Charles	First
LASTNAME	Grant	Last
MI	X	Middle Initial
GENDER	M	Gender: M or F
EMAIL	cg@successfactors.com	Email
MANAGER	ftarzanin	User ID of manager, “NO MANAGER” for no manager
HR	ditto	User ID of HR, “NO HR” for no HR
DEPARTMENT	Engineering	Department

DIVISION	Successfactors	Division
LOCATION	San Mateo	Location
GROUP	Server	Group
JOBCODE	xyz	Job Code
TIMEZONE	PST	Time Zone
DEFAULT_LOCALE	en_US	Locale
HIREDATE	01-30-2006	MM-dd-yyyy, can also be locale specific
PASSWORD	someString	Password
MATRIX_MANAGER	flim byip	Pipe separated list of matrix managers
CUSTOM_MANAGER	flim byip	Pipe separated list of custom managers
PROXY	flim byip	Pipe separated list of proxies
SECOND_MANAGER	wdoolittle	User ID of second manager
TITLE	Software Engineer	Title
BIZ_PHONE	111-222-3333	Business Phone
FAX	111-222-3333	Fax
ADDR1		Address line 1
ADDR2		Address line 2
CITY		City
STATE		State
ZIP		Zip
COUNTRY		Country
EMPID		Employee ID (on customer side)
REVIEW_FREQ		Review Frequency
LAST_REVIEW_DATE		Last Review Date
CUSTOM01		Custom fields 01 – 15
CUSTOM*		Custom fields 01 – 15. You have 15, so identification is CUSTOM01, CUSTOM02, etc through CUSTOM15.
sendWelcomeMessage	true/false	UI Equivalent: Send Welcome Message to New Users.
validateMgrHr	true/false	UI Equivalent: Validate Manager and HR fields.
managerTransfer	true/false	Automatic Manager Transfer flag
routeDocument	true/false	UI Equivalent: Automatic insertion of new manager as next document recipient if not already.
routeInboxDoc	true/false	UI Equivalent: Automatic Inbox Document Transfer To New Manager
routeEnRouteDoc	true/false	UI Equivalent: Automatic En Route Document Transfer To New Manager
routeCompletedDoc	true/false	UI Equivalent: Automatic Completed Document Copy to New Manager
managerOnlyCompany	true/false	UI Equivalent: Enforce Manager-only Implementation.
removeInProgressDocsForInactiveUsers	true/false	UI Equivalent: Remove Inactive Employees' In-Progress Documents
removeCompletedDocsForInactiveUsers	true/false	UI Equivalent: Remove Inactive Employees' Completed Documents
removeInactiveFromInProgressComp	true/false	Remove the inactive user from in-progress comp plans
removeInProgressCompForInactiveUsers	true/false	Remove the inactive users' in-progress comp plans
removeCompletedCompForInactiveUsers	true/false	Remove the inactive users' completed comp plans
removeBudgetForInactiveUsers	true/false	Remove the inactive users' completed comp plans budget
defaultPasswordField	USERID, EMAIL, FIRSTNAME, LASTNAME, RANDOM_PWD	If password is not given, use specified field as default password

The return type of the generic put method is a PutResult object. The PutResult object is documented below.

PutResult (Extended from SFObject – permissioned access)

Returned object from put method.		
Name	Type	Description
ResultCode	Integer	If put request is successful, returns 1. Otherwise, returns 0.
Errors	SFWebServiceError []	Array of SFWebServiceError objects describing errors and warnings

Handling Errors

If the PutResult object has encountered errors, the ResultCode field will be set to 0 (zero). Detailed information on the error condition will be contained in an array of SFWebServiceError objects, which have the following format.

SFWebServiceError (Extended from SFObject)

Error object		
Name	Type	Description
Code	String	Error code
Type	String	Type of Error (ie, “Warning”, Error”)
Description	String	Error description

The recommended error handling is to catch all errors (ResultCode = 0) and have an administrator review the Description in the SFWebService error and compare it to the actual User record data being imported. The administrator should consult the table below to remedy the error.

Error Descriptions and Error Numbers

The Description field of the SFWebService error will contain a textual description of the error encountered, and may reference an error number for more information. The error number can be used for more information from the table below.

For example, an error description may be:

Update failed for user: cgrant: with error: -12

Referring to the table below, we see that Error Number -12 indicates that a duplicate username was encountered. The remedy is to ensure that a unique username is used when creating a new user in the SuccessFactors system.

Error Number referenced in <description> element	Description	Impact	Remedy
-1	Internal general error.	Data in the API request may or may	Contact SuccessFactors directly to find out the

		not be updated.	root cause and find the appropriate resolution.
-6	Invalid User ID. This happens when you are trying to create a user record with a User ID that is not conforming to the SuccessFactors system's User ID standard. Please refer to this documentation on what can be used in a User ID: <LINK_TO_PM_DATA_DOCUMENTATION>.	No data will be updated.	Check if the new User ID conforms to the SuccessFactors system or not. If it doesn't please make the appropriate changes. If it does, please contact SuccessFactors to find out why you are getting this error.
-10	A cycle was found in the manager hierarchy. For example, the following would be a cycle: User1 has manager User2 and User2 has manager User1.	The Manager field will not be updated. Other fields in the request may still be updated in our system.	Check if a Manager reporting cycle has been formed when updating a user's Manager. If yes, please correct the cycle. If not, please contact SuccessFactors to find out why you are getting this error.
-11	Invalid Username. This happens when you are trying to create a user record with a Username that is not conforming to the SuccessFactors system's Username standard. Please contact SuccessFactors for a documentation on a full list of supported characters in Username.	No data will be updated.	Check if the new Username conforms to the SuccessFactors system or not. If it doesn't please make the appropriate changes. If it does, please contact SuccessFactors to find out why you are getting this error.
-12	Duplicate Username. This happens when you are trying to update a user record with a Username that is already in use by another user (active or inactive) in the system.	No data will be updated.	Check if the Username is already used by another user (active or inactive) in the SuccessFactors system. If it is, please resolve the Username conflict. If it isn't, please contact SuccessFactors to find out why you are getting this error.
-20	Matrix Manager does not exist when trying to update the Matrix Manager field for a user.	The Matrix Manager field will not be updated. Other fields in the request may still be honored and updated in our system.	Check if the Matrix Manager exists in the SuccessFactors system or not. If it doesn't exist, please import the Matrix Manager first. If it does, please contact SuccessFactors to find out why you are getting this error.
-21	Proxy User does not exist when trying to update the Proxy User field for a user.	The Proxy User field will not be updated. Other fields in the request may still be	Check if the Proxy User exists in the SuccessFactors system or not. If it doesn't exist,

		honored and updated in our system.	please import the Proxy User first. If it does, please contact SuccessFactors to find out why you are getting this error.
-22	Custom Manager does not exist when trying to update the Custom Manager field for a user.	The Custom Manager field will not be updated. Other fields in the request may still be honored and updated in our system.	Check if the Custom Manager exists in the SuccessFactors system or not. If it doesn't exist, please import the Custom Manager first. If it does, please contact SuccessFactors to find out why you are getting this error.
-23	Second Manager does not exist when trying to update the Second Manager field for a user.	The Second Manager field will not be updated. Other fields in the request may still be honored and updated in our system.	Check if the Second Manager exists in the SuccessFactors system or not. If it doesn't exist, please import the Second Manager first. If it does, please contact SuccessFactors to find out why you are getting this error.
-24	A Second Manager Cycle has been detected when trying to update the Second Manager field for a user.	The Second Manager field will not be updated. Other fields in the request may still be honored and updated in our system.	Check if there is indeed a Second Manager reporting cycle has been formed when updating a user's Second Manager. If it does, please correct the cycle. If it doesn't, please contact SuccessFactors to find out why you are getting this error.
-20	Internal general error related to Matrix Manager and/or HR Manager relationship changes.	The Matrix and/or HR Manager fields will not be updated. Other fields in the request may still be honored and updated in our system.	Please contact SuccessFactors to find out why you are getting this error.
-21	Duplicate Tag. This happens when trying to add/assign a duplicate Tag to a user.	The duplicate Tag will not be added. Other fields in the request may still be honored and updated in our system.	Check if the Tag is duplicate to the user in the SuccessFactors system. If it is, do not assign the Tag to the user. If it isn't, please contact SuccessFactors to find out why you are getting this error.
-22	The user account used for Web Services API authentication isn't validated or set up properly in our system.	No data will be updated.	Check if the user is properly set up and validated in the

			SuccessFactors system or not. If it isn't, then either find an alternative user to invoke Web Services API calls or complete the user setup and/or validation. If it does, please contact SuccessFactors to find out why you are getting this error.
--	--	--	--

Other PartnerService API Error Codes

Value in <description> element	Description	Impact	Remedy
Invalid Manager specified for user: <USERID>	Manager does not exist when trying to update the Manager field for a user.	The Manager field will not be updated. Other fields in the request may still be honored and updated in our system.	Check if the Manager exists in the SuccessFactors system or not. If it doesn't exist, please import the manager first before importing its direct reports. If it does, please contact SuccessFactors to find out why you are getting this error.
Invalid HR specified for user: <USERID>	HR does not exist when trying to update the HR field for a user.	The HR field will not be updated. Other fields in the request may still be honored and updated in our system.	Check if the HR exists in the SuccessFactors system or not. If it doesn't exist, please import the HR first. If it does, please contact SuccessFactors to find out why you are getting this error.
Failed to deactivate user: <USERID> - User does not exist	This happens when trying to deactivate a user that doesn't exist in our system.	No updates will happen in our system.	Check if the user exists in our system or not. If it doesn't, then there is no need and way to deactivate a non-existent record. If it does, please contact SuccessFactors to find out why you are getting this error.
Failed to deactivate user: <USERID>	This happens when trying to deactivate a user that is already inactive.	No updates will happen in our system.	Check if the user is active in our system. If it isn't, then you can ignore this error. If it is, please contact SuccessFactors to find out why you are getting this error.
Exception caught when routing documents for user: ...	An internal exception happened when trying to transfer documents from a former manager to the new manager for an employee.	The Manager field may or may not be updated. All other fields in the request may still be honored and updated in our system.	Please contact SuccessFactors and provide the API response to us for investigation.
Exception caught when loading user: ...	This is a generic catch-all error when trying to create or update a user.	The fields in the request may or may not be updated in our	Please contact SuccessFactors and provide the API response to us for

		system.	investigation.
Error: Missing required field for user: ...	This happens when the API request does not provide all required fields for creating or updating a user record.	The request will be rejected and no updates will happen in our system.	Review the detailed description of the error and find out what are the missing fields. Provide valid values to those missing fields and re-submit the request. If the fields do have valid values in the request, please contact SuccessFactors to find out why you are getting this error.
Field lengths are incorrect for user: ...	This happens when the API request contains certain fields whose lengths are in violation of field length restriction.	The request will be rejected and no updates will happen in our system.	Review the detailed description of the error and find out which fields are violating the field length restriction which is defined in your Data Model. You can contact SuccessFactors to find out if and what are the field length restrictions defined in your company's Data Model. Fix the violations and re-submit the request. If the fields do have valid length in the request, please contact SuccessFactors to find out why you are getting this error.

Example SOAP Request for Put UserObject

```

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance">
  <soapenv:Body>
    <ns1:put soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="PartnerService">
      <string xsi:type="xsd:string">UserObject</string>
      <SFObject soapenc:arrayType="ns2:SFObject[1]" xsi:type="soapenc:Array"
xmlns:ns2="urn:PartnerService" xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
        <SFObject href="#id0"/></SFObject>
      </ns1:put>
      <multiRef id="id0" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" xsi:type="ns3:UserObject"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/" xmlns:ns3="urn:PartnerService">
        <sfAttributes soapenc:arrayType="ns3:SFAAttribute[1]" xsi:type="soapenc:Array">
          <sfAttributes href="#id1"/>
        </sfAttributes>
        <userID xsi:type="xsd:string">cgrant_123</userID>
        <username xsi:type="xsd:string">cgrant</username>
      </multiRef>
      <multiRef id="id1" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" xsi:type="ns5:SFAAttribute"
xmlns:ns5="urn:PartnerService" xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
        <name xsi:type="xsd:string">CITY</name>

```

```

    <type xsi:type="xsd:string">String</type>
    <value xsi:type="xsd:string">San Mateo</value>
  </multiRef>
</soapenv:Body>
</soapenv:Envelope>

```

Example SOAP Response for Put UserObject – Successful

Below is a put response from a successful put UserObject call. Notice that the resultCode in the putResponse object is “1”, indicating success.

```

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <ns1:putResponse soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="PartnerService">
      <putReturn href="#id0"/>
    </ns1:putResponse>
    <multiRef id="id0" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" xsi:type="ns2:PutResult"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/" xmlns:ns2="urn:PartnerService">
      <errors xsi:type="soapenc:Array" xsi:nil="true"/>
      <lastModified xsi:type="xsd:string" xsi:nil="true"/>
      <locale xsi:type="xsd:string" xsi:nil="true"/>
      <objectId xsi:type="xsd:string" xsi:nil="true"/>
      <resultCode xsi:type="xsd:int">1</resultCode>
      <sfAttributes xsi:type="soapenc:Array" xsi:nil="true"/>
    </multiRef>
  </soapenv:Body>
</soapenv:Envelope>

```

Example SOAP Response for Put UserObject with Errors

Below is an example put response where an error was encountered. Notice that the result code in the putResponse object is “0”, indicating failure.

```

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <ns1:putResponse soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="PartnerService">
      <putReturn href="#id0"/>
    </ns1:putResponse>
    <multiRef id="id0" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" xsi:type="ns2:PutResult"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/" xmlns:ns2="urn:PartnerService">
      <errors soapenc:arrayType="ns2:SFWebServiceError[1]" xsi:type="soapenc:Array">
        <errors href="#id1"/>
      </errors>
      <lastModified xsi:type="soapenc:string" xsi:nil="true"/>
      <locale xsi:type="soapenc:string" xsi:nil="true"/>
      <objectId xsi:type="soapenc:string" xsi:nil="true"/>
      <resultCode xsi:type="xsd:int">0</resultCode>
    </multiRef>
  </soapenv:Body>
</soapenv:Envelope>

```

```
<sfAttributes xsi:type="soapenc:Array" xsi:nil="true"/>
</multiRef>
<multiRef id="id1" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xsi:type="ns3:SFWebServiceError" xmlns:ns3="urn:PartnerService"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
  <code xsi:type="soapenc:string">INTERNAL_ERROR</code>
  <description xsi:type="soapenc:string">Update failed for user: cgrant: with error: -
12</description>
  <lastModified xsi:type="soapenc:string" xsi:nil="true"/>
  <locale xsi:type="soapenc:string" xsi:nil="true"/>
  <objectId xsi:type="soapenc:string" xsi:nil="true"/>
  <sfAttributes xsi:type="soapenc:Array" xsi:nil="true"/>
  <type xsi:type="soapenc:string">Error</type>
</multiRef>
</soapenv:Body>
</soapenv:Envelope>
```