This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited. The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation shall be subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License (December 2007). Oracle America, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information on content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.
# Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Change History</td>
<td>5</td>
</tr>
<tr>
<td>Audience</td>
<td>6</td>
</tr>
<tr>
<td>Overview</td>
<td>7</td>
</tr>
<tr>
<td>Prerequisites</td>
<td>9</td>
</tr>
<tr>
<td>Common Data Types</td>
<td>11</td>
</tr>
<tr>
<td>Administrator Mode and Self-Service Mode</td>
<td>11</td>
</tr>
<tr>
<td>Application</td>
<td>12</td>
</tr>
<tr>
<td>Application Schema</td>
<td>12</td>
</tr>
<tr>
<td>Constituent</td>
<td>12</td>
</tr>
<tr>
<td>Constituent Schema</td>
<td>12</td>
</tr>
<tr>
<td>Admissions Application Web Service Operations</td>
<td>13</td>
</tr>
<tr>
<td>CREATE Application</td>
<td>13</td>
</tr>
<tr>
<td>Description</td>
<td>13</td>
</tr>
<tr>
<td>Input</td>
<td>14</td>
</tr>
<tr>
<td>Output</td>
<td>16</td>
</tr>
<tr>
<td>SAVE Application</td>
<td>18</td>
</tr>
<tr>
<td>Description</td>
<td>18</td>
</tr>
<tr>
<td>Input</td>
<td>20</td>
</tr>
<tr>
<td>Output</td>
<td>20</td>
</tr>
<tr>
<td>SUBMIT Application</td>
<td>21</td>
</tr>
<tr>
<td>Description</td>
<td>21</td>
</tr>
<tr>
<td>Input</td>
<td>24</td>
</tr>
<tr>
<td>Output</td>
<td>25</td>
</tr>
<tr>
<td>GET Application</td>
<td>26</td>
</tr>
<tr>
<td>Description</td>
<td>26</td>
</tr>
<tr>
<td>Input</td>
<td>27</td>
</tr>
<tr>
<td>Output</td>
<td>28</td>
</tr>
<tr>
<td>GET Applications</td>
<td>29</td>
</tr>
<tr>
<td>Description</td>
<td>29</td>
</tr>
<tr>
<td>Input</td>
<td>30</td>
</tr>
<tr>
<td>Output</td>
<td>31</td>
</tr>
<tr>
<td>GET Attachment</td>
<td>32</td>
</tr>
<tr>
<td>Description</td>
<td>32</td>
</tr>
<tr>
<td>Input</td>
<td>33</td>
</tr>
<tr>
<td>Output</td>
<td>34</td>
</tr>
<tr>
<td>User Account Web Service Operations</td>
<td>36</td>
</tr>
<tr>
<td>CREATE Account</td>
<td>36</td>
</tr>
<tr>
<td>Description</td>
<td>36</td>
</tr>
<tr>
<td>Input</td>
<td>37</td>
</tr>
<tr>
<td>Output</td>
<td>38</td>
</tr>
<tr>
<td>AUTHENTICATION</td>
<td>39</td>
</tr>
<tr>
<td>Description</td>
<td>39</td>
</tr>
<tr>
<td>Input</td>
<td>40</td>
</tr>
<tr>
<td>Output</td>
<td>40</td>
</tr>
<tr>
<td>Payment Integration Web Service Operations</td>
<td>42</td>
</tr>
<tr>
<td>INITIATE PAYMENT</td>
<td>42</td>
</tr>
<tr>
<td>Description</td>
<td>42</td>
</tr>
<tr>
<td>Input</td>
<td>44</td>
</tr>
<tr>
<td>Output</td>
<td>45</td>
</tr>
<tr>
<td>COMPLETE PAYMENT</td>
<td>47</td>
</tr>
<tr>
<td>Description</td>
<td>47</td>
</tr>
<tr>
<td>Input</td>
<td>49</td>
</tr>
<tr>
<td>Output</td>
<td>50</td>
</tr>
<tr>
<td>List of Values Web Service Operations</td>
<td>52</td>
</tr>
<tr>
<td>GET List of Values</td>
<td>52</td>
</tr>
<tr>
<td>Description</td>
<td>52</td>
</tr>
</tbody>
</table>
Input ............................................................................................................................................ 55
Output .......................................................................................................................................... 56
Configuration Hints and Tips .................................................................................................................... 58
Constituent Transaction Manager ......................................................................................................... 58
Recommendations: .............................................................................................................................. 58
Data Update Rules ............................................................................................................................... 58
Recommendations: .............................................................................................................................. 60
Entity Registry ................................................................................................................................. 61
Recommendations: .............................................................................................................................. 61
Application Configuration .................................................................................................................... 61
Recommendations: .............................................................................................................................. 61
List of Values ...................................................................................................................................... 62
Recommendations: .............................................................................................................................. 62
Appendix .................................................................................................................................................... 63
WSDL ................................................................................................................................................... 63
Admissions Application Web Services ............................................................................................ 63
User Account Web Service .............................................................................................................. 64
List Of Values Web Service .............................................................................................................. 65
Sample Code ......................................................................................................................................... 65
Calling SaveApplication – Adobe Flex .............................................................................................. 65
Change History

<table>
<thead>
<tr>
<th>Release</th>
<th>Change Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bundle 21 (April, 2011)</td>
<td>Changes to the SAD_FAULT_RESP message example (page 15).</td>
</tr>
</tbody>
</table>
| Additional Features July 2011 (Bundle 22) | - Updated the List of Values Web Service Operations section to account for the new LOV Context field.  
                                             - Updated Configuration Hints and Tips for ‘List of Values’ and ‘Entity Registry’ sections.  
                                             - Updated the Appendix section with new diagrams and code. |

This updated guide is available on My Oracle Support as part of the documentation delivered for the releases indicated in the table.
**Audience**

This document is aimed for developers who want to access and use the:

- Admission Applications Web Service (AAWS).
- Get List of Values, User Creation, and User Authentication web service operations.

Technical consultants should use this document to learn about the technologies used by these web services and to determine the scope of data that the web services can exchange between the online application user interface that the academic institution builds and the PeopleSoft system. Business and functional analysts should use this document to understand how the web services function.

This document is aimed for users who are familiar with:

- A tool or technology to build and deploy the online application user interface.
- PeopleCode.
- Web services concepts, mainly XML, SOAP, and WSDL.
- PeopleSoft Integration Broker.Campus Solutions Recruiting and Admissions functionality.

The following topics are covered in this document:

- The **Common Data Types** chapter provides information about the data that is exchanged between the online application and the PeopleSoft system.
- The **Admission Applications Web Service (AAWS) Operations** chapter describes the six web service operations for creating, saving, viewing, and submitting admission applications and attachments.
- The **User Account Web Service Operations** chapter describes the New User Registration and User Authentication web service operations.
- The **List Of Values Web Service Operation** chapter describes the Get List of Values web service operation.

Each section that describes a web service operation contains:

- An overview of the service operation.
- Steps that a web service operation performs to execute input parameters and give an output.
- Conditions that will make the service throw an error message.
- Examples of XML inbound and outbound messages that a service operation can accept or produce.
Overview

Admission Applications Web Service (AAWS)

Different academic institutions have different needs for capturing admission application data. Also, application data can vary within the same academic institution between various campuses, departments, careers, and types of applicants. For example, application data for international and domestic applicants can vary within the same institution. This creates the need for institutions to develop their own online application user interface that can interact with the Campus Solutions system. With AAWS, institutions can connect their online application with the Campus Solutions system.

The AAWS feature, initially delivered with Feature Pack 3, offers access to the Campus Solutions Recruiting and Admissions functionality through service operations. The feature offers the following service operations:

- Create Application
- Save Application
- Submit Application
- Get Application
- Get Applications
- Get Attachment

In addition, the following three service operations were delivered in Feature Pack 3:

- User Registration
- User Authentication
- Get List of Values

You can use the additional three services to:

- Provide usernames and passwords for access to your online application.
- Authenticate users who want to retrieve a saved application for further edits and submission.
- Retrieve the list of values from the Campus Solutions system for data entry fields on the online application.

Feature Pack 5 delivers the following two service operations:

- Initiate Payment
- Complete Payment

Any online application user interface that is web-service enabled and SOAP compliant can access these web services.

Note: The User Registration, User Authentication, and Payment Services service operations are delivered for only supporting online applications. These web services are not intended to be used for any other purpose.

Extensibility

Academic institutions can modify the Campus Solutions tables to capture additional data from the applicants. For example, if an institution wants to include a data entry field titled Job Title in Previous Employment on their online application, they can use the Entity Registration component to include such a data capture element in Campus Solutions.

See Also

- PeopleSoft Enterprise Campus Solutions 9.0 Admissions Application Web Service Developer's Guide
- Recruiting and Admissions PeopleBook Documentation for FP 3

Bulk Loading of Applications

Academic institutions can use the File Parser feature to import admission applications from third-party vendors or other sources into the staging tables. Refer to the PeopleSoft Enterprise Campus Solutions 9.0 Admissions Application Web Service Developer's Guide.
Application Web Service Developer's Guide for information about this feature.
Prerequisites

AAWS can be used by end user devices (such as a browser or a PDA), enterprise applications or any other third party software that can interface through standards based means.

AAWS can be used by:

- Oracle presentation technologies (such as PORTAL, Application Development Framework [or ADF], and PeopleSoft components and pages).
- Oracle middleware (such as Enterprise Service Bus and BPEL).
- Oracle Applications (such as Enterprise, EnterpriseOne, and E-Business Suite).
- Third party presentation technologies.
- Third party middleware and applications.

AAWS is constructed upon Oracle PeopleSoft Enterprise PeopleTools 8.49. Oracle’s PeopleSoft Enterprise PeopleTools 8.49 and higher versions support the tools and technologies listed in the following table.

<table>
<thead>
<tr>
<th>Technology/Product</th>
<th>Version/Specification</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apache Xalan</td>
<td>1.9. C++ version</td>
<td>Apache Xalan is an XSLT processor for transforming XML documents into HTML, text, or any other document type.</td>
</tr>
<tr>
<td>Apache Xerces</td>
<td>2.6.0. C++ version</td>
<td>Apache Xerces is an XML parser.</td>
</tr>
<tr>
<td>AS2</td>
<td>RFC 3335</td>
<td>Based on software product IP*Works! EDI V6.</td>
</tr>
<tr>
<td>FTP/FTPS</td>
<td>5.1</td>
<td>NA</td>
</tr>
<tr>
<td>HTTP</td>
<td>1.1</td>
<td>NA</td>
</tr>
<tr>
<td>HTTPS</td>
<td>SSL 3.0</td>
<td>NA</td>
</tr>
<tr>
<td>iWay SOAPswitch</td>
<td>5.5.3</td>
<td>NA</td>
</tr>
<tr>
<td>JMS</td>
<td>1.0.2</td>
<td>NA</td>
</tr>
<tr>
<td>Oracle XSL Mapper</td>
<td>10.1.2.02</td>
<td>NA</td>
</tr>
<tr>
<td>SOAP</td>
<td>1.1 and 1.2</td>
<td>All SOAP output from PeopleSoft Integration Broker is version 1.1. PeopleSoft Integration Broker understands versions 1.1 and 1.2. AAWS uses Schema for the SOAP/1.1 envelope, and SOAP/1.2 envelope found at <a href="http://schemas.xmlsoap.org/soap/envelope/">http://schemas.xmlsoap.org/soap/envelope/</a>.</td>
</tr>
<tr>
<td>UDDI</td>
<td>2.0</td>
<td>NA</td>
</tr>
<tr>
<td>WSDL</td>
<td>1.1</td>
<td>AAWS uses the schema found at <a href="http://schemas.xmlsoap.org/wsd1">http://schemas.xmlsoap.org/wsd1</a>.</td>
</tr>
<tr>
<td>Technology/Product</td>
<td>Version/Specification</td>
<td>Comments</td>
</tr>
<tr>
<td>----------------------</td>
<td>-----------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>WS-Addressing</td>
<td>1.1</td>
<td>Currently the WS-Addressing document is a formal submission to W3C, the current submission is using schema under <a href="http://schemas.xmlsoap.org/ws/2004/08/addressing/">http://schemas.xmlsoap.org/ws/2004/08/addressing/</a>. However, the integration gateway uses an older version, <a href="http://schemas.xmlsoap.org/ws/2003/03/addressing/">http://schemas.xmlsoap.org/ws/2003/03/addressing/</a>.</td>
</tr>
<tr>
<td>WS-I Basic Profile</td>
<td>1.0</td>
<td>The WS-I Basic Profile 1.0 specification mandates support for SOAP 1.1, WSDL 1.1, UDDI 2.0, HTTP 1.1, SSL 3.0 (or HTTPS), XML 1.0 (2nd Edition), and XML Schema (Part 1 and 2).</td>
</tr>
<tr>
<td>WS-Interoperability</td>
<td>1.2.1</td>
<td>NA</td>
</tr>
<tr>
<td>XML</td>
<td>1.0 (Second edition.)</td>
<td>NA</td>
</tr>
<tr>
<td>XML schema</td>
<td>Parts 1 and 2</td>
<td>NA</td>
</tr>
<tr>
<td>XSLT</td>
<td>1.0</td>
<td>Use of Apache Xerces 2.6.0 and Apache Xalan 1.9, XSLT 1.0 is supported.</td>
</tr>
<tr>
<td>XQuery, XPath</td>
<td>1.0</td>
<td>Used in PeopleSoft Integration Broker transform programs. Use of Apache Xerces 2.6.0 and Apache Xalan 1.9, XQuery and XPath 1.0 is supported.</td>
</tr>
</tbody>
</table>
Common Data Types

This chapter lists the data types that AAWS uses to exchange data between the Campus Solutions system and the online application.

Administrator Mode and Self-Service Mode

Administrator Mode can be used for the six operation services tied to the SAD_ADMISSIONS service:

- SAD_CREATEAPPL
- SAD_GETAPPL
- SAD_GETAPPLS
- SAD_GETATTACH
- SAD_SAVEAPPL
- SAD_SUBMITAPPL

Same thing is true for the service operation tied to the SCC_LOV service:

- SCC_GET_LOV

When used, the message request contains the following tag: <SCC_ADMIN_MODE>.

For example, the request message for the service operation SAD_CREATEAPPL is as follow:

```xml
<?xml version="1.0"?>
<SAD_CREATEAPPL_REQ xmlns="http://xmlns.oracle.com/Enterprise/HCM/services">
  <SCC_ADMIN_MODE>
    <INSTITUTION>PSUNV</INSTITUTION>
    <ADM_APPL_CTR>BUSN</ADM_APPL_CTR>
    <COUNTRY></COUNTRY>
    <CITIZENSHIP_STATUS></CITIZENSHIP_STATUS>
    <PREFERENCES>
      <PREFERENCE>
        <INSTITUTION>PSUNV</INSTITUTION>
        <ACAD_CAREER>BUSN</ACAD_CAREER>
        <ACAD_PROG>GRFIN</ACAD_PROG>
        <ACAD_PLAN>FINANC-MBA</ACAD_PLAN>
        <ACAD_SUB_PLAN></ACAD_SUB_PLAN>
        <ADMIT_TYPE>FYR</ADMIT_TYPE>
      </PREFERENCE>
    </PREFERENCES>
  </SCC_ADMIN_MODE>
</SAD_CREATEAPPL_REQ>
```
<ADMIT_TERM>0630</ADMIT_TERM>
</PREFERENCE>
</PREFERENCES>
</SAD_CREATEAPPL_REQ>

Note: The service operations tied to the User Registration service (SCC_USERREG) do not support Administrator Mode since only the self-service users can authenticate or create a user account for them self.

Application
The Application is the core object containing the complete view of an admission application. The Application object is derived from the complete set of data available within the Campus Solutions admissions core data model.

Application Schema
Refer to the Application.xsd PDF attachment to view the XML XSD for an admissions application.

Constituent
The Constituent object represents the core information associated with the person using AAWS. The person could be an applicant or an administrator using the online application or an administrator using the Campus Solutions Recruiting and Admissions functionality. The Constituent object is derived from the complete set of person related data available in both the Campus Solutions and Human Resources person data models.

Constituent Schema
AAWS uses the Constituent schema to track the person related information for an application.

Refer to the Constituent.xsd PDF attachment to view the XML XSD for a constituent.
Admissions Application Web Service Operations

Service Name: SAD_ADMISSIONS

Admissions Application Web Service (AAWS) offers functionality in the following areas:

1. Admission Applications:

   Users, such as applicants, can use an online application user interface to:
   - Create admission applications in the Campus Solutions system.
   - Save, update, or submit the application.

2. Application Attachments:

   Online application users can associate attachments of various formats with an admission application. Administrators can track these attachments using the Application Transactions and Maintain Application components.

---

CREATE Application

Description

<table>
<thead>
<tr>
<th>Service:</th>
<th>SAD_ADMISSIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operation:</td>
<td>SAD_CREATEAPPL: Create an Initialized Application</td>
</tr>
<tr>
<td>Summary:</td>
<td>Allows the user to create an admission application in the Campus Solutions system. Using this service may result in one or more applications being created. It can also result in multiple applicant preferences being created within the same application, for example, entry of another program or plan for an existing application is a preference.</td>
</tr>
</tbody>
</table>
### Description:
The Create Application service operation call results in one or more admission applications being initialized. The application center settings on the Application Configuration component control whether multiple applications (or preferences) can be created. For example, if you do not have an application center set up for an applicant who selects an Undergraduate career, Liberal Arts program, and History plan, the service throws an error.

The service operation populates the admission application fields based on the settings on the User Defaults and Installation - AD components. At application creation time the operation does not physically save the data to the admissions staging tables. The save process does not occur until a later point in time when the user selects a save or submit option on the online application. Therefore, this operation is more concerned with the initialization of an application, driven by configuration, rather than creating a physical application in the database.

### Users:
- An applicant using an online application.
- An administrator using an online application for creating an application on behalf of an applicant who has or does not have an EMPLID.

### Processing:
This service operation performs the following steps:

1. Validate that the user is authenticated (and is not a Guest)
2. If Administrator mode then:
   - Validate the required input administrator parameters.
   - Validate that the administrator is authorized to access the application center, institution, career, program and plan.
3. Validate the required input parameters for application creation.
4. Determine the transaction that must be used to process the Institution/Academic Career combination. Define transactions on the Transaction Setup page.
5. Begin a new transaction for the request. Assign a new TEMP_ID to the constituent.
6. Using the supplied input parameters, look up the Application configuration component settings to determine how many application preferences should be created for this request.
7. Return response though the output parameters.

### Output:
After processing, the operation creates an admission application with default data and preferences. The user can subsequently update and submit this application through the Save and Submit service operations.

### Error Conditions:
The following conditions result in a service error:

- User is not authenticated (for example, if an anonymous Guest tries to sign into the online application, the service throws an error message).
- Administrative user is not authorized to access the application center, institution, career, program or plan.
- Required input parameters have not been supplied by the user (see Figure 1: SAD_CREATEAPPL_REQ Message Parameters and the Common Data Types section).

---

**Input**

**Message:** SAD_CREATEAPPL_REQ

The following diagram shows the input parameters that the SAD_CREATEAPPL service operation receives.
from an online application:

Figure 1: SAD_CREATEAPPL_REQ Message Parameters

The following input parameters are mandatory for the online application to pass to the service operation:

Non-administrator (Applicant) mode:

- Institution
- Academic Career
- Admit Term
- Academic Program
- Academic Plan

Administrator mode:

- Institution
- Admission Application Center
- Academic Career
- Admit Term
- Academic Program
- Academic Plan
- Country and Citizenship Status IF your application mapping requires them.
The following is an example of the SAD_CREATEAPPL_REQ message that the SAD_CREATEAPPL service operation receives from an online application:

```xml
<?xml version="1.0"?>
<SAD_CREATEAPPL_REQ>
  <COUNTRY>AUS</COUNTRY>
  <PREFERENCE>
    <INSTITUTION>PSUNV</INSTITUTION>
    <ACAD_CAREER>UGRD</ACAD_CAREER>
    <ACAD_PROG>AA</ACAD_PROG>
    <ACAD_PLAN>ARTHIST</ACAD_PLAN>
    <ADMIT_TYPE>FYR</ADMIT_TYPE>
    <ADMIT_TERM>0747</ADMIT_TERM>
  </PREFERENCE>
</SAD_CREATEAPPL_REQ>
```

**Output**

**Message: SAD_CREATEAPPL_RESP**

When the Integration Broker receives the SAD_CREATEAPPL_REQ message, it responds with the SAD_CREATEAPPL_RESP message.

The following diagram shows the output parameters that the SAD_CREATEAPPL service operation passes to the online application:

![Diagram](image)

**Figure 2: SAD_CREATEAPPL_RESP Message Parameters**

Refer to the Common Data Types section for the parameters in the ADM_APPL_DATA (Application) and Constituent data types.

The following is an example of the SAD_CREATEAPPL_RESP message that the SAD_CREATEAPPL service operation transmits to the online application:

```xml
<?xml version="1.0"?>
<SAD_CREATEAPPL_RESP>
  <PREFERENCE>
    <ADM_APPL_DATA> <!-- Application data shape -->
    </ADM_APPL_DATA>
  </PREFERENCE>
  <CONSTITUENT> <!-- Constituent data shape -->
  </CONSTITUENT>
</SAD_CREATEAPPL_RESP>
```

**Fault: SAD_FAULT_RESP**
If the service operation encounters an error condition, it responds with the SAD_FAULT_RESP message.

The following is an example of the SAD_FAULT_RESP message that the SAD_CREATEAPPL service operation transmits to the online application:

```xml
<?xml version="1.0" encoding="UTF-8"?>
  <SOAP-ENV:Body>
    <SOAP-ENV:Fault>
      <faultcode>300</faultcode>
      <faultstring>A Error occurred processing this request</faultstring>
      <detail>
        <MSGS>
          <MSG>
            <ID>14200-556</ID>
            <DESCR>Validation error in preference PSUNV/UGRD/AA/MUSIC/VOIC/FYR/0690</DESCR>
            <PROPS/>
          </MSG>
          <MSG>
            <ID>14200-556</ID>
            <DESCR>Tag ACAD_SUB_PLAN has an invalid value VOIC. Check and try again.</DESCR>
            <PROPS><SCC_ENTITY_INST_ID>PREFERENCE ROW 1</SCC_ENTITY_INST_ID><PROPNAME>ACAD_SUB_PLAN</PROPNAME></PROPS>
          </MSG>
        </MSGS>
      </detail>
    </SOAP-ENV:Fault>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```
### SAVE Application

**Description**

<table>
<thead>
<tr>
<th>Service:</th>
<th>SAD_ADMISSIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operation:</td>
<td>SAD_SAVEAPPL: Save an Application (for Later Submission)</td>
</tr>
<tr>
<td>Summary:</td>
<td>Allows the user to save an admissions application to the Campus Solutions system. The service operation initially posts the application to the admissions staging tables. Applications in the staging tables may at some point of time become eligible to be posted to the admissions production tables.</td>
</tr>
<tr>
<td>Description:</td>
<td>The Save Application service call results in the given admissions application being saved to the Campus Solutions system. The application may be partially complete when the online application user decides to use a Save option that triggers a call to this service operation. The service operation saves the application using an Update Via Replacement approach; in other words, the SAD_SAVEAPPL operation deletes any pre-existing application (based on a unique key) for the applicant and then saves the new (or current) version of the application. When saving an application, the operation assigns an application number (ADM_APPL_NBR) to the application for tracking. The application must pass through validation checks before it gets saved. The service operation validates all aspects of the application data including data entered and translate values selected by the user on the online application, and custom business validation rules that the institution has incorporated into the online application. The service operation saves the valid applications to the admissions staging tables. If the constituent (person) who initiated the application save process is known (for example, the constituent has an EMPLID), then the constituent portion of the application is immediately eligible to be posted to the Campus Solutions and HCM person production tables. Otherwise, the constituent data remains in the staging tables until addressed by an administrator. In any case, the application data itself is saved to the staging tables and is not posted to admissions production tables until a later submit occurs. Applications may include attachments. The online application must encode the attachment in Base64 format before passing the attachment to Campus Solutions.</td>
</tr>
<tr>
<td>Users:</td>
<td>• An applicant using an online application. • An administrator using an online application for creating an application on behalf of an applicant or a student.</td>
</tr>
</tbody>
</table>
### Processing:

This service operation performs the following steps:

1. Validate that the user is authenticated (and is not a Guest)
2. If administrator mode, then:
   a. Validate the required input administrator parameters.
   b. Validate that the administrator is authorized to access the application center, institution, career, program and plan.
   c. Use the EMPLID supplied by the administrator for the transaction.
3. Validate that the applicant is authorized to save the application (for example, verify whether the applicant is the same person who created the application in the call to the SAD_CREATEAPPL service operation).
4. If administrator mode, then use the input parameter Application Center value for the application.
5. If applicant mode, then use the input parameters and the settings on the Application Configuration component to assign an application center to the application.
6. To enter the Application Citizenship details, use the applicant details in the input parameters, if necessary.
7. To enter the Constituent Institution (needed for Data Update Rules), use the Admissions Application Institution input parameter value.
8. Validate the constituent and application data using the Constituent.xsd and Application.xsd files.
9. Associate the TEMP_ID (assigned during the SAD_CREATEAPPL processing) with the application.
10. Save the validated constituent and application data to the staging tables.
11. Add the assigned ADM_APPL_NBR to the response message.
12. If the constituent is not known to the Campus Solutions system (for example, the applicant does not have an EMPLID):
   a. Run the Search Match process according to the Search Match settings on the Transaction Setup component.
   b. Add or update a HCM person record according to Search Match configuration and apply the necessary data update rules during this processing.
13. Create or update a prospect record in the production tables according to the settings on the Application Configuration component.

### Output:

After completing the processing steps, the service operation:

1. Posts the valid application data to the Campus Solutions admissions staging tables.
2. Posts the valid constituent data to the Transaction Manager constituent staging tables.
3. Optionally, creates or updates a HCM person record for a valid application.
4. Optionally, creates a prospect record for a known applicant.

### Error Conditions:

The following conditions result in a service error:

- User is not authenticated (for example, if an anonymous Guest tries to sign into the online application, the service throws an error message).
- User is not authorized to access the application (for example, if the applicant is trying to access an application created by another applicant).
- Application is invalid (that is, the application fails validation processing).
- Required input parameters have not been supplied by the user (see Figure 3: SAD_SAVEAPPL_REQ Message Parameters and the Common Data Types section).
Input

Message: SAD_SAVEAPPL_REQ

The following diagram shows the input parameters that the SAD_SAVEAPPL service operation receives from an online application:

![Diagram showing input parameters]

Figure 3: SAD_SAVEAPPL_REQ Message Parameters

Refer to the Common Data Types section for the parameters in the ADM_APPL_DATA (Application) and Constituent data types.

The following input parameters are mandatory for the online application to pass to the service operation:

Non-administrator (Applicant) mode:
- ADM_APPL_DATA
- CONSTITUENT

Administrator mode:
- Institution
- Admission Application Center
- ADM_APPL_DATA
- CONSTITUENT

The following is an example of the SAD_SAVEAPPL_REQ message that the SAD_SAVEAPPL service operation receives from an online application:

```xml
<?xml version="1.0"?>
<SAD_SAVEAPPL_REQ>
  <ADM_APPL_DATA>
    <!-- Application data shape -->
  </ADM_APPL_DATA>
  <CONSTITUENT>
    <!-- Constituent data shape -->
  </CONSTITUENT>
</SAD_SAVEAPPL_REQ>
```

Output

Message: SAD_SAVEAPPL_RESP

When the Integration Broker receives the SAD_SAVEAPPL_REQ message, it responds with the
SAD_SAVEAPPL_RESP message.

The following diagram shows the output parameters that the SAD_SAVEAPPL service operation passes to the online application:

![Figure 4: SAD_SAVEAPPL_RESP Message Parameters](image)

Refer to the Common Data Types section for the parameters in the Constituent data type.

The following is an example of the SAD_SAVEAPPL_RESP message that the SAD_CREATEAPPL service operation transmits to the online application:

```xml
<?xml version="1.0"?>
<SAD_SAVEAPPL_RESP>
  <ADM_APPL_NBR>00023456</ADM_APPL_NBR>
</SAD_SAVEAPPL_RESP>
```

Fault: SAD_FAULT_RESP

Refer to the SAD_FAULT_RESP message example in the CREATE Application section.

---

**SUBMIT Application**

**Description**

<table>
<thead>
<tr>
<th>Service:</th>
<th>SAD_ADMISSIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operation:</td>
<td>SAD_SUBMITAPPL: Submit a Completed Application</td>
</tr>
<tr>
<td>Summary:</td>
<td>Allows the user to submit a completed admissions application to the Campus Solutions system. The service operation initially saves the applications to the admissions staging tables and subsequently posts the staged application data to the admissions production tables.</td>
</tr>
</tbody>
</table>
**Description:** The Submit Application service call results in the given admissions application being submitted to the Campus Solutions system. The application must be complete (according to the institution's data capture requirements). The online application user interface and the underlying Campus Solutions data model ensure that the submitted admission application is complete.

The application submit process involves first validating the data, then saving the data to the admissions staging tables, and finally transferring the staged data into the admissions production tables.

When initially saving the application to the staging tables, if the application does not have an application number, the service operation assigns an application number (ADM_APPL_NBR) to the application for tracking purpose. The operation does not assign an application number if another operation (for example, the SAD_SAVEAPPL operation) has already assigned an application number.

The admission application must pass through validation checks before it gets submitted. The service operation validates all aspects of application data including data entered and translate values selected by the user on the online application and custom business validation rules that the institution has incorporated into the online application.

If the constituent who initiated the application submit process is known to the Campus Solutions system (for example, the constituent has an EMPLID), then the constituent and application data is eligible to be posted to the Campus Solutions admissions and person production tables. Otherwise, the application remains in the staging tables until addressed by an administrator.

When configured to do so, the constituent can pay an application fee and/or request an application fee waiver.

Applications may include attachments. The online application must encode the attachment in Base64 format before passing the attachment to Campus Solutions.

**Users:**
- An applicant using an online application.
- An administrator using an online application for creating an application on behalf of an applicant or a student.
<table>
<thead>
<tr>
<th>Processing</th>
<th>This service operation performs the following steps:</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Validate that the user is authenticated (and is not a Guest)</td>
</tr>
</tbody>
</table>
| 2.         | If administrator mode then:  
|            | a. Validate the required input administrator parameters.  
|            | b. Validate that the administrator is authorized to access the application center, institution, career, program and plan.  
|            | c. Use the EMPLID supplied by the administrator for the transaction. |
| 3.         | Validate that the applicant is authorized to submit the application (for example, verify whether the applicant is the same person who created the application in a call to the SAD_CREATEAPPL service operation.) |
| 4.         | If administrator mode, then use the input parameter Application Center value for the application. |
| 5.         | If applicant mode, then use the input parameters and the settings on the Application Configuration component to assign an application center to the application. |
| 6.         | To enter the Application Citizenship details use the applicant details in the input parameters, if necessary. |
| 7.         | To enter the Constituent Institution (needed for data update rules) use the Admissions Application Institution input parameter value. |
| 8.         | Validate the constituent and application data using the Constituent.xsd and Applicant.xsd files. |
| 9.         | Associate the TEMP_ID (assigned during the SAD_CREATEAPPL processing) with the application. |
| 10.        | Save the validated constituent and application data to the staging tables. |
| 11.        | If the constituent is not known to the Campus Solutions system (for example, the applicant does not have an EMPLID):  
|            | a. Run the Search Match process according to the Search Match settings on the Transaction Setup component.  
|            | b. Add or update a HCM Person record according to Search Match configuration and apply the necessary data update rules during this processing. |
| 12.        | If Fee payment is required but has not yet been paid:  
|            | a. Call the Student Financials Fee Calculation API to calculate the fee amount.  
|            | b. If Fee Waivers are allowed then allow the application to progress to submitted status  
|            | c. If Fee Waivers are Not allowed then  
|            | 1. Set the application data to Saved Status  
|            | 2. Add the Fee details to the response message  
|            | 3. The caller must handle this response by initiating the payment services. Note it is possible for Fee calculation to result in a zero (0) amount in which case the caller should not initiate the payment service. |
| 13.        | Create or update a prospect record in the production tables according to settings on the Application Configuration component.  
|            | If the constituent is known (for example, the applicant has an EMPLID), post the application data into the admissions production tables and apply the necessary data update rules during this processing. |
Output: After completing the processing steps, the service operation:

1. Posts the valid application data to the Campus Solutions admissions staging tables.
2. Posts the valid constituent data to the Transaction Manager constituent staging tables.
3. Transfers the application and constituent data from the staging tables to the Campus Solutions person and admissions production tables.
4. Optionally, creates or updates a HCM person record for a valid application.
5. Optionally, creates a prospect record for a known applicant.

Error Conditions: The following conditions result in a service error:

- User is not authenticated (for example, an if an anonymous Guest user tries to sign into the online application, the service operation throws an error message.)
- User is not authorized to access the application (for example, the applicant is trying to access an application created by another applicant).
- Application is invalid (that is, the application fails validation processing).
- Required input parameters have not been supplied by the user (see Figure 5: SAD_SUBMITAPPL_REQ Message Parameters and the Common Data Types section).

Input Message: SAD_SUBMITAPPL_REQ

The following diagram shows the input parameters that the SAD_SUBMITAPPL service operation receives from an online application:

Figure 5: SAD_SUBMITAPPL_REQ Message Parameters

Refer to the Common Data Types section for the parameters in the ADM_APPL_DATA (Application) and Constituent data types.

The following input parameters are mandatory for the online application to pass to the service operation:

Non-administrator (Applicant) mode:

- ADM_APPL_DATA
- CONSTITUENT
Administrator mode:

- Institution
- Admission Application Center
- ADM_APPL_DATA
- CONSTITUENT

The following is an example of the SAD_SUBMITAPPL_REQ message that the SAD_SUBMITAPPL service operation receives from an online application:

```xml
<?xml version="1.0"?>
<SAD_SUBMITAPPL_REQ>
  <ADM_APPL_DATA>
    <!-- Application data shape -->
  </ADM_APPL_DATA>
  <CONSTITUENT>
    <!-- Constituent data shape -->
  </CONSTITUENT>
</SAD_SUBMITAPPL_REQ>
```

**Output**

**Message: SAD_SUBMITAPPL_RESP**

When the Integration Broker receives the SAD_SUBMITAPPL_REQ message, it responds with the SAD_SUBMITAPPL_RESP message.

The following diagram shows the output parameters that the SAD_SUBMITAPPL service operation passes to the online application:

![Diagram](image)

**Figure 6: SAD_SUBMITAPPL_RESP Message Parameters**

Refer to the Common Data Types section for the parameters in the Constituent data type.

The following is an example of the SAD_SUBMITAPPL_RESP message that the SAD_SUBMITAPPL service operation transmits to the online application:

```xml
<?xml version="1.0"?>
<SAD_SUBMITAPPL_RESP>
  <ADM_APPL_NBR>00023456</ADM_APPL_NBR>
  <FEE>0023456</FEE>
</SAD_SUBMITAPPL_RESP>
```
Fault: SAD_FAULT_RESP

Refer to the SAD_FAULT RESP message example in the CREATE Application section.

---

**GET Application**

**Description**

<table>
<thead>
<tr>
<th>Service:</th>
<th>SAD_ADMISSIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operation:</td>
<td>SAD_GETAPPL: Retrieve a Single Application</td>
</tr>
<tr>
<td>Summary:</td>
<td>Allows the user to retrieve a specific admissions application from the Campus Solutions system. The application may exist in the admissions staging tables or in the admissions production tables. Saved applications reside in the staging tables and submitted applications reside in the production tables.</td>
</tr>
<tr>
<td>Description:</td>
<td>Use this service operation to retrieve a saved or submitted application using the ADM_APPL_NBR key. An online application can obtain the ADM_APPL_NBR application key by calling the SAD_GETAPPLS service, which retrieves a summary of all the applications associated with the current authenticated user. The SAD_GETAPPL operation can operate under applicant or administrative modes. When using applicant mode, the application data returned is restricted to those applications entered previously by the authenticated online application user. When using administrator mode, the application data returned from the admissions production tables is subject to application center row level security.</td>
</tr>
<tr>
<td>Users:</td>
<td>• An applicant using an online application. • An administrator using an online application to retrieve data that they previously saved or submitted. The administrator can retrieve an application for an applicant who has or does not have an EMPLID.</td>
</tr>
</tbody>
</table>
Processing: This service operation performs the following steps:

1. Validate that the user is authenticated (and is not a Guest).
2. If administrator mode, then:
   a. Validate the required input administrator parameters.
   b. Validate that the administrator is authorized to access the application center.
   c. Use the EMPLID supplied by the administrator for the transaction.
3. Validate that the applicant is authorized to retrieve the application (for example, verify whether the applicant is the same person who created the application).
4. If administrator mode, then use the EMPLID input parameter to retrieve the application.
5. If applicant mode, then use the EMPLID input parameter associated with the authenticated user profile to retrieve the application.
6. To retrieve the application based on the ADM_APPL_NBR key:
   a. First, check whether the application exists in the admissions staging tables using the SCC_TEMP_ID key.
   b. If the application does not exist in the staging tables, check the admissions production tables using the EMPLID key.
7. Return the application data, if found.

Output: After completing the processing steps, the service operation returns a single application if it exists in the staging or production tables. All the application details along with constituent details are returned, if the application exists.

Error Conditions: The following conditions result in a service error:

- User is not authenticated (for example, if an anonymous Guest user tries to sign into the online application, the service operation throws an error message).
- User is not authorized to access the application (for example, the applicant is trying to access an application created by another user).
- Required input parameters have not been supplied by the user (see Figure 7: SAD_GETAPPL_REQ Message Parameters).
- Administrative user is not authorized to access the application due to the application center security configuration.

Input
Message: SAD_GETAPPL_REQ

The following diagram shows the input parameters that the SAD_GETAPPL service operation receives from an online application:
Figure 7: SAD_GETAPPL_REQ Message Parameters

Refer to the Common Data Types section for the parameters in the ADM_APPL_DATA (Application) and Constituent data types.

The following input parameters are mandatory for the online application to pass to the service operation:

Non-administrator (Applicant) mode:

- Academic Career
- Application Number

Administrator mode:

- Institution
- Admission Application Center

The following is an example of the SAD_GETAPPL_REQ message that the SAD_GETAPPL service operation receives from an online application:

```xml
<?xml version="1.0"?>
<SAD_GETAPPL_REQ>
  <ADM_APPL_NBR>00024656</ADM_APPL_NBR>
  <ACAD_CAREER>UGRD</ACAD_CAREER>
</SAD_GETAPPL_REQ>
```

Output Message: SAD_GETAPPL_RESP

When the Integration Broker receives the SAD_GETAPPL_REQ message, it responds with the SAD_GETAPPL_RESP message.

The following diagram shows the output parameters that the SAD_GETAPPL service operation passes to the online application:

![Diagram showing output parameters]

Figure 8: SAD_GETAPPL_RESP Message Parameters

The following is an example of the SAD_GETAPPL_RESP message that the SAD_GETAPPL service operation transmits to the online application:

```xml
<?xml version="1.0"?>
<SAD_GETAPPL_RESP>
  <ADM_APPL_DATA>
    <!-- Application data shape -->
  </ADM_APPL_DATA>
  <CONSTITUENT>
    <!-- Constituent data shape -->
  </CONSTITUENT>
</SAD_GETAPPL_RESP>
```

Fault: SAD_FAULT_RESP
Refer to the SAD_FAULT_RESP message example in the CREATE Application section.

### GET Applications

**Description**

<table>
<thead>
<tr>
<th>Service:</th>
<th>SAD_ADMISSIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operation:</td>
<td>SAD_GETAPPLS: Retrieve a Summary of Applications</td>
</tr>
<tr>
<td>Summary:</td>
<td>Allows the user to retrieve a summary of all active admission applications from the Campus Solutions system for the current authenticated user. The applications summary includes applications found in both the admissions staging and production database tables.</td>
</tr>
<tr>
<td>Description:</td>
<td>Use this service operation to retrieve a summary of all the applications that reside in the Campus Solutions staging and production tables. The service operation retrieves only those applications that the authenticated user is authorized to access. SAD_GETAPPLS can operate under applicant or administrative modes. When using administrator mode, the application data returned from the Admissions production tables is subject to application center row level security. When using applicant mode, the application data returned is restricted to those applications entered previously by the authenticated online application user.</td>
</tr>
</tbody>
</table>
| Users:        | • An applicant using an online application.  
• An administrator using an online application to retrieve data that he or she previously saved or submitted. The administrator can retrieve an application for an applicant who has or does not have an EMPLID. |
Processing

This service operation performs the following steps:

1. Validate that the user is authenticated (and is not a Guest).
2. If administrator mode then:
   a. Validate the required input administrator parameters.
   b. Validate that the administrator is authorized to access the application center.
   c. Use the EMPLID supplied by the administrator for the transaction.
3. Determine the list of configured application transactions from the Transaction Setup component.
4. Retrieve all the application transactions for all the configured application transactions.
5. Retrieve all admission applications from the Admissions staging tables that the authenticated user is authorized to access.
6. If applicant mode, then retrieve all the applications associated with the authenticated EMPLID from the production tables. To retrieve the application based on the ADM_APPL_NBR key:
   a. First, check whether the application exists in the admissions staging tables using the SCC_TEMP_ID key.
   b. If the application does not exist in the staging tables, check the admissions production tables using the EMPLID key.
8. Return the Application Summary, which indicates ID (EMPLID or SCC_TEMP_ID), Application Number, Institution, Academic Career and Application Status (SAVED, SUBMITTED, ERROR, or POSTED).

Output: After processing the steps, a summary of all active applications is returned.

Error Conditions: The following conditions result in a service error:

- User is not authenticated (for example, if an anonymous Guest user tries to sign into the online application, the service operation throws an error message).
- Required input parameters have not been supplied by the user (see Figure 9: SAD_GETAPPLS_REQ Message Parameters).

**Input**

**Message:** SAD_GETAPPLS_REQ

The following diagram shows the input parameters that the SAD_GETAPPLS service operation receives from an online application:

![Input Diagram]

Figure 9: SAD_GETAPPLS_REQ Message Parameters

The following input parameters are mandatory for the online application to pass to the service operation:

Non-administrator (Applicant) mode:

No Required parameters
Administrator mode:

- Institution
- Admission Application Center
- Academic Program

The following is an example of the SAD_GETAPPLS_REQ message that the SAD_GETAPPLS service operation receives from an online application:

```xml
<?xml version="1.0"?>
<SAD_GETAPPLS_REQ/>
```

**Output**

**Message: SAD_GETAPPLS RESP**

When the Integration Broker receives the SAD_GETAPPLS_REQ message, it responds with the SAD_GETAPPLS_RESP message.

The following diagram shows the output parameters that the SAD_GETAPPLS service operation passes to the online application:

![Diagram showing output parameters]

**Figure 10: SAD_GETAPPLS_RESP Message Parameters**

The following is an example of the SAD_GETAPPLS_RESP message that the SAD_GETAPPLS service operation transmits to the online application:

```xml
<?xml version="1.0"?>
<SAD_GETAPPLS_RESP>
 <APPLICATION>
   <ADM_APPL_NBR>00024656</ADM_APPL_NBR>
   <INSTITUTION>PSUNV</INSTITUTION>
   <ACAD_CAREER>UGRD</ACAD_CAREER>
   <APPL_STATUS>DRAFT</APPL_STATUS>
 </APPLICATION>
 <APPLICATION>
   <ADM_APPL_NBR>00024657</ADM_APPL_NBR>
   <INSTITUTION>PSUNV</INSTITUTION>
   <ACAD_CAREER>UGRD</ACAD_CAREER>
   <APPL_STATUS>SUBMITTED</APPL_STATUS>
 </APPLICATION>
</SAD_GETAPPLS_RESP>
Fault: SAD_FAULT_RESP

Refer to the SAD_FAULT_RESP message example in the CREATE Application section.

---

## GET Attachment

**Description**

<table>
<thead>
<tr>
<th>Service:</th>
<th>SAD_ADMISSIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operation:</td>
<td>SAD_GETATTACH: Retrieve an Attachment</td>
</tr>
<tr>
<td>Summary:</td>
<td>Allows the user to retrieve the byte content of an admissions application attachment from the Campus Solutions system.</td>
</tr>
<tr>
<td>Description:</td>
<td>Use this service operation to retrieve an attachment associated with the admission application. The online application must encode the attachment byte content in Base64 format when a user saves or submits the application data through AAWS. Subsequently, the online application can use SAD_GETATTACH to retrieve the attachment. After retrieval, the online application must decode the attachment. The applicant or administrator can then use the appropriate file type viewer to view the attachment. The attachment is stored within the Campus Solutions attachment framework in a database table. You can use the Application Transactions or Maintain Applications components to access the application attachments.</td>
</tr>
<tr>
<td>Users:</td>
<td>• An applicant using an online application. • An administrator using an online application to retrieve data that he or she previously saved or submitted. The administrator can retrieve an application (and the associated attachments) for an applicant who has or does not have an EMPLID.</td>
</tr>
</tbody>
</table>
### Processing

This service operation performs the following steps:

1. Validate that the user is authenticated (and is not a *Guest*).
2. If administrator mode, then:
   a. Validate the required input administrator parameters.
   b. Validate that the administrator is authorized to access the application center.
   c. Use the EMPLID supplied by the administrator for the transaction.
3. Verify whether the applicant is authorized to retrieve the attachment (for example, verify whether the applicant is the same person who originally created the attachment through AAWS).
4. If administrator mode, then use the EMPLID input parameter to retrieve the attachment.
5. If applicant mode, then use the EMPLID input parameter associated with the authenticated User Profile to retrieve the attachment.
6. To retrieve the attachment:
   a. First, check whether the attachment exists in the staging tables using the SCC_TEMP_ID key.
   b. If the attachment does not exist in the staging tables, check the production tables using the EMPLID key.
7. Return the attachment, if found.

### Output:

After completing the processing steps, if the attachment exists, the service operation returns the attachment data to the online application.

### Error Conditions:

The following conditions result in a service error:

- User is not authenticated (for example, if an anonymous *Guest* user tries to sign into the online application, the service operation throws an error message).
- User is not authorized to retrieve the attachment (for example, the applicant is trying to access an attachment created by another user).
- Required input parameters have not been supplied by the user (see Figure 11: SAD_GETATTACH_REQ Message Parameters). Administrative user is not authorized to access the attachment due to application center configuration.

Note: The Online Application may issue multiple calls to the SAD_GETATTACH service operation in order to retrieve multiple Application attachments.

**Input**

**Message:** SAD_GETATTACH_REQ

The following diagram shows the input parameters that the SAD_GETATTACH service operation receives from an online application:
Figure 11: SAD_GETATTACH_REQ Message Parameters

The following input parameters are mandatory for the online application to pass to the service operation:

Non-administrator (Applicant) mode:

- Attachment Sequence Number
- Academic Career
- Application Number

Administrator mode:

- Institution
- Admission Application Center

The following is an example of the SAD_GETATTACH_REQ message that the SAD_GETATTACH service operation receives from an online application:

```xml
<?xml version="1.0"?>
<SAD_GETATTACH_REQ>
  <ACAD_CAREER>UGRD</ACAD_CAREER>
  <ADM_APPL_NBR>00024501</ADM_APPL_NBR>
  <ATTACH_SEQ_NBR>1</ATTACH_SEQ_NBR>
</SAD_GETATTACH_REQ>
```

Output Message: SAD_GETATTACH_RESP

When the Integration Broker receives the SAD_GETATTACH_REQ message, it responds with the SAD_GETATTACH_RESP message.

The following diagram shows the output parameter that the SAD_GETATTACH service operation passes to the online application:

Figure 12: SAD_GETATTACH_RESP Message Parameters

The following is an example of the SAD_GETATTACH_RESP message that the SAD_GETATTACH service
operation transmits to the online application:

```xml
<?xml version="1.0"?>
<SAD_GETATTACH_RESP>
  <ADM_APPL_ATCH class="R">
    <EMPLID>2</EMPLID>
    <ACAD_CAREER>UGRD</ACAD_CAREER>
    <STDNT_CAR_NBR>0</STDNT_CAR_NBR>
    <ADM_APPL_NBR>00024501</ADM_APPL_NBR>
    <ATTACH_SEQ_NBR>1</ATTACH_SEQ_NBR>
    <DESCR254>test file</DESCR254>
    <SCC_ROW_ADD_OPRID/>
    <SCC_ROW_ADD_DTTM/>
    <SCC_ROW_UPD_OPRID/>
    <SCC_ROW_UPD_DTTM/>
    <ATTACHSYSFILENAME>ADM_APPL_ATCH_20090331161239_Who Columns.doc</ATTACHSYSFILENAME>
    <ATTACHUSERFILE>Who Columns.doc</ATTACHUSERFILE>
    <AV_ATCH_TYPE/>
    <BASE64FILE>0M8R4KGxGuEAAAAAAIAAAAAAAAAAPgADAP7/CqAGAAAAAAA</BASE64FILE>
  </ADM_APPL_ATCH>
  <CONSTITUENT>
    <!-- Constituent data shape -->
  </CONSTITUENT>
</SAD_GETATTACH_RESP>
```

**Fault:** SAD_FAULT_RESP

Refer to the SAD_FAULT_RESP message example in the CREATE Application section.
### User Account Web Service Operations

**Service Name:** *SCC_USERREG*

Feature Pack 3 of the Campus Solutions User Account Web Service offers functionality in the following areas:

- **Account Creation**
  
  The SCC_USERREG_CREATEACCT service operation can create a new PeopleTools user profile. The service operation provides support for the PeopleTools user management system. In addition, the service operation supports other user management systems through the delivered adapter architecture. You need to create a new adapter to have your PeopleSoft system interact with another user management system through SCC_USERREG.

  Use this service operation to enable an online application user to create a username and password.

- **Authentication (Logon)**
  
  Use the SCC_USERREG_AUTHENTICATE service operation to authenticate an online application user.

  This service operation uses the delivered User Manager adapter to validate a User ID and password combination with the installed user management system. The service operation is delivered with the ability to authenticate a user using the PeopleTools user management system.

### CREATE Account

**Description**

<table>
<thead>
<tr>
<th>Service:</th>
<th>SCC_USERREG</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operation:</td>
<td>SCC_USERREG_CREATEACCT: <em>Create an Account</em></td>
</tr>
<tr>
<td>Summary:</td>
<td>Allows the user to create a new account (PeopleTools user profile) in the configured user management system. The user can specify both the username and the password when requesting a user account.</td>
</tr>
</tbody>
</table>
Description: A call to the Create Account service operation results in none or a single user account being created in the configured user management system. The service operation is delivered with the standard PeopleTools user management system.

The service operation communicates to the user management system through an adapter based architecture. Therefore, it is possible to support integration with a variety of customized user management or Identity Management (IDM) systems through the creation of new adapters.

When calling this service operation, the user needs to choose a username and password. If the username is valid and does not already exist in the user management system, then the service operation creates a user account. The new user account is created using the SCC_SS_TEMPLATE user account. The service operation copies the user specific information, such as the username, into this template to create the new account. Therefore, you must set up the SCC_SS_TEMPLATE user account to allow access to AAWS (based on role and permission security). For information on setting up SCC_SS_TEMPLATE, refer to the AAWS Developer’s Guide.

Users:
- An applicant using an online application.
- An administrator using an online application.

Processing
This service operation performs the following steps:

1. Verify whether the input parameters have been passed by the online application (username, password and confirm password).
2. Retrieve any constituent information from the input parameters and validates the information.
3. Call the user management adapter to:
   a. Verify that the requested username does not exist in the user management system.
   b. Create the new user account.
   c. Authenticate the user account and signs the user into the online application.
5. Prepare the response message.

Output: The service operation either does not create a user account or it creates a new user account which the applicant or administrator can immediately use.

Error Conditions: The following conditions result in a service error:
- The username, password or confirm password input parameters are invalid.
- Constituent information provided through the input parameters is invalid.
- A user account for the same username already exists in the user management system.

**Input**

**Message:** SCC_UP_CREATEACCT_REQ

The following diagram shows the input parameters that the SCC_USERREG_CREATEACCT service operation receives from an online application:
The following input parameters are mandatory for the online application to pass to the service operation:

- SCC_USERNAME
- SCC_PASSWORD
- SCC_CONFIRMPWD
- CONSTITUENT

The following is an example of the SCC_UR_CREATEACCT_REQ message that the SCC_USERREG_CREATEACCT service operation receives from an online application:

```xml
<?xml version="1.0"?>
<SCC_UR_CREATEACCT_REQ>
  <SCC_USERNAME>KANGA</SCC_USERNAME>
  <SCC_PASSWORD>Rooly23</SCC_PASSWORD>
  <SCC_CONFIRMPWD>Rooly23</SCC_CONFIRMPWD>
  <CONSTITUENT>
    <!-- Constituent data shape -->
  </CONSTITUENT>
</SCC_UR_CREATEACCT_REQ>
```

**Output Message:** SCC_UR_CREATEACCT_RESP

When the Integration Broker receives the SCC_UR_CREATEACCT_REQ message, it responds with the SCC_UR_CREATEACCT_RESP message.

The following diagram shows the output parameter that the SCC_USERREG_CREATEACCT service operation passes to the online application:

**Figure 14: SCC_UR_CREATEACCT_RESP Message Parameters**

Refer to the Common Data Types section for the parameters in the Constituent data type.

The following is an example of the SCC_UR_CREATEACCT_RESP message that the SCC_USERREG_CREATEACCT service operation transmits to the online application:

```xml
<?xml version="1.0"?>
<SCC_UR_CREATEACCT_RESP>
```

**Figure 13: SCC_UR_CREATEACCT_REQ Message Parameters**

Refer to the Common Data Types section for the parameters in the Constituent data type.

The following input parameters are mandatory for the online application to pass to the service operation:
**Fault:** SAD_FAULT_RESP

Refer to the SAD_FAULT_RESP message example in the CREATE Application section.

---

**AUTHENTICATION**

**Description**

<table>
<thead>
<tr>
<th>Service:</th>
<th>SCC_USERREG</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operation:</td>
<td>SCC_USERREG_AUTHENTICATE: Authenticate Account Credentials (Logon)</td>
</tr>
<tr>
<td>Summary:</td>
<td>Allows an existing user account to be authenticated. After authentication, the service operation signs the user into the PeopleSoft system. The signed in user can then access features, such as AAWS.</td>
</tr>
<tr>
<td>Description:</td>
<td>The SCC_USERREG_AUTHENTICATE service operation call verifies whether the supplied username and password combination is correct using the configured user management system.</td>
</tr>
</tbody>
</table>

If the username and password combination is correct, then the user is signed into the PeopleSoft system.

All users accessing the SCC_USERREG_AUTHENTICATE operation have an Anonymous status until they have been successfully authenticated.

Anonymous users have minimal system access and **do not** have access to features, such as AAWS.

Use SCC_USERREG_AUTHENTICATE to allow a user to access your online application through a “front door”, such as a Sign In page. Once signed in, the PeopleSoft system continues to authenticate the user for each AAWS request. This check, once signed in, is done by the PeopleSoft system and not by SCC_USERREG_AUTHENTICATE.

| Users: | • An applicant using an online application.  
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>• An administrator using an online application.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Processing:</th>
<th>This service operation performs the following steps:</th>
</tr>
</thead>
</table>
| 1. Verify whether the input parameters exist (username, password).  
2. Call the user management adapter to validate the username and password combination.  
3. Use the Transaction Setup settings to retrieve any known information on the authenticated constituent.  
4. Prepare the response. |

| Output: | User is signed into the PeopleSoft system. |
Error Conditions: The following condition results in an error from the service:

- The input parameters, username and password, are invalid.
- The username and password combination cannot be authenticated using the user management system.

**Input**

**Message:** SCC_UR_AUTHENTICATE_REQ

The following diagram shows the input parameters that the SCC_USERREG_AUTHENTICATE service operation receives from an online application:

![Input Diagram](image)

**Figure 15: SCC_UR_AUTHENTICATE_REQ Message Parameters**

The following input parameters are mandatory for the online application to pass to the service operation:

- Username
- Password

The following is an example of the SCC_UR_AUTHENTICATE_REQ message that the SCC_USERREG_AUTHENTICATE service operation receives from an online application:

```xml
<?xml version="1.0"?>
<SCC_UR_AUTHENTICATE_REQ>
  <SCC_USERNAME>KANGA</SCC_USERNAME>
  <SCC_PASSWORD>Rooly23</SCC_PASSWORD>
</SCC_UR_AUTHENTICATE_REQ>
```

**Output**

**Message:** SCC_UR_AUTHENTICATE_RESP

When the Integration Broker receives the SCC_UR_AUTHENTICATE_REQ message, it responds with the SCC_UR_AUTHENTICATE_RESP message.

The following diagram shows the output parameter that the SCC_USERREG_AUTHENTICATE service operation passes to the online application:

![Output Diagram](image)

**Figure 16: SCC_UR_AUTHENTICATE_RESP Message Parameters**

The following is an example of the SCC_UR_AUTHENTICATE_RESP message that the SCC_USERREG_AUTHENTICATE service operation transmits to the online application:

```xml
<?xml version="1.0"?>
<SCC_UR_AUTHENTICATE_RESP>
  
</SCC_UR_AUTHENTICATE_RESP>
```
Fault: SAD_FAULT_RESP

Refer to the SAD_FAULT_RESP message example in the CREATE Application section.
Payment Integration Web Service Operations

INITIATE PAYMENT

Description

<table>
<thead>
<tr>
<th>Service: SSF_EPAYMENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operation: SSF_INITIATE_EPAYMENT: <em>Initiate a Payment</em></td>
</tr>
</tbody>
</table>

Summary: Allows the user to record a newly initiated electronic payment in Student Financials and provides access to Campus Solutions’ hosted payment framework. The hosted payment framework provides information necessary to transition the user to a designated third party payment provider’s hosted site to make a payment. The payment will not be posted to the General Ledger until the Complete ePayment service (SSF_COMPLETE_EPAYMENT) is invoked successfully.

This web service is adapter-based. AAWS has been delivered with its own custom Payment Application adapter and will perform its own validation according to Admissions Application functional requirements.

Note that for the AAWS FP5 release, the Initiate ePayment service will operate only within Self Service mode (and cannot be used within Administrative mode).
| Description: | The Initiate ePayment service call will create a new SF Payment in Student Financials in Initiated Status, assigning a new SF Payment Reference Number (SF_PMT_REF_NBR) for tracking. This service uses the payment application’s own tracking number as the merchant reference number passed to the third party payment processor. For AAWS, that tracking number is the application number (ADM_APPL_NBR). The application number will be stored in the SF Payment as ADM_APPL_NBR and SSF_EXT_TRANS_REF and the SF Payment Reference number will be stored in the Admissions Tender Staging table. In addition, the Temp ID will be stored in the SF Payment and Log. The service, through the AAWS adapter, will create new Admissions Tender Staging records (if they do not exist) for all tenders associated with the application center used in the creation of the application.

The Initiate ePayment service will perform initial validation. For instance, the payment method must be either credit card or electronic check (CC or EC), the amount has to be greater than zero, and the Payment Application must be defined in the system table SSF_ELEC_PMTS.

When the AAWS adapter is invoked, it will perform further validation on the transaction.

- The payment amount and currency must match the previously calculated application fee stored in the Admissions Application Staging record (SAD_APL_DAT_STG). This amount should have been previously calculated in and returned from the SUBMIT web service.
- The transaction must have been originated by the owner of the application.
- The application must be in Saved (SV) status.
- If the payment is for a credit card tender type, the SF Merchant must be defined in the Application Center associated with the application. For electronic check transactions, the Electronic Check Merchant ID must be defined in the Application Center. These SF Merchants must be associated with Payment Merchants that are defined as “hosted”.
- The tender used must be defined as a valid tender on the Application Fee code associated with the Application Center.

| Users: | An applicant using an online application. |
### Processing:
This service operation performs the following steps:

1. Validate the parameters passed in to the web service.
2. Create an SF Payment to record the transaction. The SF Payment will be in Initiated status. For AAWS, SF Payment Detail records, which will later be used to record the transaction in the General Ledger, will be based on the Application Fee Item Type setup.
3. Call the AAWS adapter to:
   - Validate the application
   - Validate the payment parameters
   - Create Application Tender Staging records
   - Save the application
4. Call the hosted payment framework. The hosted payment framework will:
   - Based on the hosted payment adapter associated with the Payment Merchant, perform an initial web service callout to the third party to pass demographic information about the applicant, such as name, address, email and phone. Note that this information may or may not be used, depending on the third party payment processor. In addition, a shared secret may be passed to the third party to authenticate electronic check transactions. For AAWS, this shared secret is the Constituent birth date. The third party will return a temporary token that identifies the transaction to both the third party and Campus Solutions.
   - Using the HTML object definition on the Payment Merchant, create a string that will be used to either http post or redirect to the third party.

### Output:
After completing the processing steps, the service operation:

6. Records the transaction in the SF Payment table and Detail records.
7. Creates new Admissions Application Tender Staging records.
8. Optionally, perform a web service callout to the third party payment processor to pass demographic information and receive a token identifying the transaction.

### Error Conditions:
The following conditions result in a service error:

- The payment method is invalid.
- The payment amount is zero.
- The Payment Application is invalid.
- Required input parameters have not been supplied by the user
- The transaction does not pass the AAWS adapter validation.

### Input
**Message:** SSF_INITIATE_EPAYMENT_REQ

The following diagram shows the input parameters that the SSR_INITIATE_EPAYMENT_REQ service operation receives from an online application:
Figure 17: SSF_INITIATE_EPAYMENT_REQ Message Parameters

The following input parameters are mandatory for the online application to pass to the service operation:

- Payment Application
- Payment Application Key (Application Number)
- Payment Amount
- Currency Code
- Payment Method (Credit card or electronic check)

The following is an example of the SSF_INITIATE_EPAYMENT_REQ message that the SSF_INITIATE_EPAYMENT service operation receives from an online application:

```xml
<?xml version="1.0"?>
<SSF_INITIATE_EPAYMENT_REQ
xmlns="http://xmlns.oracle.com/Enterprise/HCM/services">
  <SCC_PMT_APPL>SADAAWS</SCC_PMT_APPL>
  <SCC_PMT_APPL_KEY>00024828</SCC_PMT_APPL_KEY>
  <SCC_EPAY_TOTAL_AMT>20</SCC_EPAY_TOTAL_AMT>
  <CURRENCY_CD>USD</CURRENCY_CD>
  <SCC_EPAY_PAYMETHOD>CC</SCC_EPAY_PAYMETHOD>
</SSF_INITIATE_EPAYMENT_REQ>
```

Output Message: SSF_INITIATE_EPAYMENT_RESP

When the Integration Broker receives the SSF_INITIATE_EPAYMENT_REQ message, it responds with the SSF_INITIATE_EPAYMENT_RESP message.

The following diagram shows the output parameter that the SSF_INITIATE_EPAYMENT service operation passes to the online application:

Figure 18: SSF_INITIATE_EPAYMENT_RESP Message Parameters
The following is an example of the SSF_INITIATE_EPAYMENT_RESP message that the SSR_INITIATE_EPAYMENT service operation transmits to the online application:

```xml
<?xml version="1.0"?>
<SSF_INITIATE_EPAYMENT_RESP
xmlns="http://xmlns.oracle.com/Enterprise/HCM/services">
<SCC_REDIRECT>N</SCC_REDIRECT>
<SCC_HTML_STRING><![CDATA[<SCRIPT LANGUAGE='JavaScript'>
writeConsole('https://198.136.211.18:8443/upay/web/index.jsp',
'\input type="HIDDEN" name="TICKET_NAME" value="SADAAWS"><input type="HIDDEN" name="UPAY_SITE_ID" value="89"><input type="HIDDEN" name="TICKET" value="9ed5ac65-7b5b-40f4-bf4c-300634d4e33d">');
function writeConsole(bind1, bind2)
{
    top.consoleRef=window.open('', 'Payment');
    top.consoleRef.focus;
    top.consoleRef.document.writeln('<html><head><title>Payment</title><
</head> '+
'BODY onload="submitForm"><form name="payform" action="'+bind1+'" method="POST"><'+bind2+'<SCRIPT LANGUAGE="JavaScript" onload="submitForm()"><\'+
'/'+
'function submitForm(){document.payform.submit()}'+
'</'+
'/'+
'function writeCookie(name, value, days, childWindow)
{
    if(days){
        (time = new Date()).setTime(new
Date().getTime()+days*24*60*60*1000);
        var exp = '\'; expires='+time.toGMTString();
    }else
    {
        var exp='';
    }
    childWindow.document.cookie=name+'='+value+exp+'; path=/";
}
function readCookie(name)
{
    var cookies = document.cookie.split(';');
    for(var i=0; i<cookies.length; i++)
    {
        var cookie=cookies[i].replace(/\s+/, '');
        if (cookie.indexOf(name+'=')==0) return
        cookie.substring(name.length+1);
    }
    return null;
}
</SCRIPT>]]>
</SCC_HTML_STRING>
<SCC_HPAY_INTG_ID>QA_CS_MODEL1A_AAWS</SCC_HPAY_INTG_ID>
</SSF_INITIATE_EPAYMENT_RESP>

Fault: SSF_TRANS_FAULT_RESP
If the service operation encounters an error condition, it responds with the SSF_TRANS_FAULT_RESP message.

The following is an example of the SSF_TRANS_FAULT_RESP message that the SSF_INITIATE_EPAYMENT service operation transmits to the online application:

```xml
<?xml version="1.0"?>
  <SOAP-ENV:Body>
    <SOAP-ENV:Fault>
      <faultcode>Client</faultcode>
      <faultstring>An Error occurred processing this request (14098,286)</faultstring>
      <detail>
        <MSGS>
          <MSG>
            <ID>14871-11</ID>
            <DESCR>The Payment Amount must be greater than zero. (14871,11)</DESCR>
          </MSG>
        </MSGS>
      </detail>
    </SOAP-ENV:Fault>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

### COMPLETE PAYMENT

**Description**

<table>
<thead>
<tr>
<th>Service:</th>
<th>SSF_EPAYMENT_TRANS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operation:</td>
<td>SF_COMPLETE_EPAYMENT: Complete Electronic Payment</td>
</tr>
<tr>
<td>Summary:</td>
<td>This web service completes a previously initiated (SSF_INITIATE_EPAYMENT) SF Payment, by authorizing the transaction, posting to the General Ledger and submitting the application. This web service should be invoked after the transaction is submitted by the user from the third party payment processor user interface.</td>
</tr>
</tbody>
</table>
The Complete ePayment web service interprets the response from the third party payment processor by invoking the hosted payment framework. The data received from the third party is transformed into Campus Solutions data elements to allow the update of both Student Financials tables and the calling Payment Application (AAWS).

If the user submitted the transaction successfully from the third party processor, this web service will complete any processing required in Student Financials to record the payment. The session id, which is a unique token generated and returned by the third party to identify the payment, as well as the last 4 digits of the credit card or account number, will be stored in the SF Payment table from within the hosted payment framework.

The ePayment API will be invoked to authorize the transaction. If successful, the payment will be processed in Student Financials by setting the SF Payment Status to Complete (SF_PAYMENT_STATUS = B), creating the SF Payment Log with the results of the authorization, and eventually deleting the SF Payment.

The transaction will be recorded directly in the General Ledger using the new Direct to GL API (as opposed to posting to a Student Financials account).

Finally, the adapter for AAWS will be invoked to submit the application. The application fee status will be set to Received, the Application Fee Paid amount and GL Sent Amount will be set to the Payment Amount. The Admissions Application Tender Staging record’s tender amount will also be set to the Payment Amount.

If the user cancelled the transaction from the third party payment processor’s hosted site, the SF Payment status will be set to Cancelled. No update will occur on the AAWS adapter and the applicant can submit another payment against the application.

Users:
- An applicant using an online application.

Processing:
This service operation performs the following steps:

1. Validate the parameters passed in to the web service.
2. Call the Hosted Payment framework using the Hosted Payment adapter passed into the web service. The hosted payment framework will interpret the name/value pairs and transform them into Campus Solutions data elements using the hosted payment adapter settings. It will extract the Payment Application ID and Key (application number) to correctly identify the transaction. It will update the session id and last 4 digits of the account or credit card number in the SF Payment.
3. Complete the SF Payment associated with the application. For successfully submitted transactions, the transaction will be authorized and sent to the General Ledger.
4. Call the AAWS adapter to update the proper fields in the application and submit it.

Output:
After completing the processing steps, the service operation:

1. Sets the SF Payment status to Complete or Cancelled according to the user action on the third party processor. It will set the status to Declined if authorization fails. The SF Payment will be deleted and the SF Payment Log will be created to record the transaction results.
2. Creates the General Ledger Transaction records for later processing to the General Ledger.
3. Submits the application if the payment was successfully authorized.
Error Conditions: The following conditions result in a service error:

- The Hosted Payment Adapter ID is invalid.
- The Return URL type is invalid (Valid values are S, C or E)
- The Payment Application is invalid.
- Required input parameters have not been supplied by user.
- The transaction does not pass the AAWS adapter validation.

Input

Message: SSF_COMPLETE_EPAYMENT_REQ

The following diagram shows the input parameters that the SSF_COMPLETE_EPAYMENT_REQ service operation receives from an online application:

```xml
<?xml version="1.0"?>
<SSF_COMPLETE_EPAYMENT_REQ
xmlns="http://xmlns.oracle.com/Enterprise/HCM/services"
<SCC_HPAY_INTG_ID>QA_CS_MODEL1A_AAWS</SCC_HPAY_INTG_ID>
<SCC_EPAY_URL_TYPE>S</SCC_EPAY_URL_TYPE>
<SCC_HP_RESPONSE_VALUES>
    <SCC_TP_PARM>
        <name>UPAY_SITE_ID</name>
        <value>89</value>
    </SCC_TP_PARM>
    <SCC_TP_PARM>
        <name>EXT_TRANS_ID</name>
        <value>SADAAWS_00024828</value>
    </SCC_TP_PARM>
    <SCC_TP_PARM>
        <name>Another</name>
        <value>Close this Window</value>
    </SCC_TP_PARM>
    <SCC_TP_PARM>
        <name>PSHome</name>
        <value>ps</value>
    </SCC_TP_PARM>
    <SCC_TP_PARM>
        <name>Portal</name>
```

Figure 19: SSF_COMPLETE_EPAYMENT_REQ Message Parameters

The following input parameters are mandatory for the online application to pass to the service operation:

- Hosted Payment Adapter ID
- Return URL Type (Success, Cancelled, Error)
- Hosted Payment Response Values (Name/Value pairs returned from third party)

The following is an example of the SSF_COMPLETE_EPAYMENT_REQ message that the SSF_COMPLETE_EPAYMENT service operation receives from an online application:
When the Integration Broker receives the SSF_COMPLETE_EPAYMENT_REQ message, it responds with the SSF_COMPLETE_EPAYMENT_RESP message.

The following diagram shows the output parameter that the SSF_COMPLETE_EPAYMENT service operation passes to the online application:

**Figure 20: SSF_COMPLETE_EPAYMENT_RESP Message Parameters**
The following is an example of the SSF_COMPLETE_EPAYMENT_RESP message that the SSF_COMPLETE_EPAYMENT service operation transmits to the online application:

```xml
<?xml version="1.0"?
<SSF_COMPLETE_EPAYMENT_RESP
xmlns="http://xmlns.oracle.com/Enterprise/HCM/services">
  <SF_PMT_REF_NBR>000000000043</SF_PMT_REF_NBR>
  <SCC_EPAY_MERCH_REF>00024828</SCC_EPAY_MERCH_REF>
  <CR_CARD_DIGITS>1111</CR_CARD_DIGITS>
  <SF_POST_STATUS>S</SF_POST_STATUS>
  <SF_PAYMENT_STATUS>B</SF_PAYMENT_STATUS>
  <SSF_PMT_SOURCE_STATUS>S</SSF_PMT_SOURCE_STATUS>
  <SSF_PMT_MSG>Your payment was successfully processed.</SSF_PMT_MSG>
  <SSF_APPL_MSG>Your application 00024828 was successfully submitted.</SSF_APPL_MSG>
</SSF_COMPLETE_EPAYMENT_RESP>
```

**Fault:** SSF_TRANS_FAULT_RESP

Refer to the SSF_TRANS_FAULT_RESP message example in the Initiate ePayment section.
List of Values Web Service Operations

Service Name: **SCC_LOV**

Most self-service user interfaces (UIs) constructed outside of your PeopleSoft Campus Solutions system will contain data fields that have predefined or “prompt” values used to control and streamline data entry for the applicant. To use list of values that are defined inside your PeopleSoft Campus Community system, use the List of Values Web Service (SCC_GET_LOV, introduced in Feature Pack 3, July 2010). The service recognizes and provides the values that are used to populate drop-down list boxes on a user interface and for validating selections. The required parameters are the record and the field names you want list of values to be returned. The service operation looks, by default, at the prompt table defined in Application Designer for the desired record and field name to retrieve the values for a field. Optionally, if you want to alter the default way in which the operation picks up the list of values for a field, use the List of Values Setup component.

See Also
PeopleSoft Enterprise Campus Community Fundamentals 9.0 PeopleBook, “Setting Up List of Values”.

---

### GET List of Values

**Description**

<table>
<thead>
<tr>
<th>Service:</th>
<th>SCC_LOV</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operation:</td>
<td>SCC_GET_LOV: <em>Get List of Values</em></td>
</tr>
<tr>
<td>Summary:</td>
<td>Allows a user interface built outside of your PeopleSoft Campus Community system to request a list of prompt or translate field values for a given field that are stored inside the PeopleSoft Campus Community system. The list of values is defined in the PeopleSoft database or in the List of Values Setup page. This list of values is used in populating drop-down list boxes or check boxes on an online application user interface or is used for validating a value selected by the user.</td>
</tr>
</tbody>
</table>
Description: The Get List of Values service operation call allows the online application to request a list of values for a field from the PeopleSoft system. After retrieval, the online application user selects one of these values for the field.

When calling the service operation, the online application nominates the record and the field for which the list of values is to be returned.

The online application can request one or more lists of values in a single Get List of Values call. Therefore, we highly recommend that your online application should request all the lists of values or most of the list of values at initialization time. Subsequently, call this service operation again whenever necessary.

The Get List of Values service operation also allows the user to label each list of values request with a unique identifier (see below).

The List of Values Setup component supports both applicant (Self Service mode) and administrative access to this service operation.

Applicant access is unrestricted, that is, the service operation returns all the database values for a list of values request. In the case of applicant access, you can use the List of Values Setup component to exclude specific values from the returned list of values.

Administrative access is restricted by academic structure security settings. Therefore, an administrator using this service will only see values for which they are authorized (as defined in the List of Values setup and the academic structure security settings (for example, institution, career, program, and plan security).

You can use the Get List of Values service operation to populate a list of values on a pre-Login page. For example, a prelogin page may ask applicants to select an academic career before signing into the online application. Therefore, a user does not have to be authenticated to use this service operation. The List of Values Setup component has been provided for the typical data capture fields that you will deploy on an online application.

Users:
- An applicant using an online application.
- An administrator using an online application.
Processing | This Service Operation performs the following steps:
---|---
1. Determine if a List of Values setting exists for the record, name and LOV Context (optional) combination passed inside the LOV request message.
   
   If a LOV Context is passed, it uses the LOV settings defined for the LOV Context, record and name passed inside the LOV request message.
   
   If the LOV Context is not passed, it uses the LOV settings defined for the *Default* LOV Context, record and field name passed inside the LOV request message.
2. Determine whether the online application is calling the operation in Applicant (Self-Service mode) or Administrator mode.
3. Get the list of values record name, field name, LOV Context (optional) and any keys from the input parameters.
4. Look up the Source value on the List of Values Setup component.
   
   If the source is set to *Request XML*, the online application needs to pass the data entered by the user for the fields (mostly the table keys) set with *Request XML*. The online application will then need to provide the list of values service with all required keys in the request XML. This allows the PeopleSoft system to Lookup the defined prompt using the keys supplied by the online application. For example, INSTITUTION may be defined as a key with source: *Request XML*. The online application will need to supply, for example, a value of *Undergraduate* when calling the List of Values service operation.
   
   If the source is set to *Constant* and a constant value is set up, the service operation will retrieve the list of values that the online application can display for the field and, in addition, assign the constant value as the default value for the field. For example, suppose your online application has a required Admit Term field and you want the online application to load with a value of *Fall 2010* automatically selected. The applicant can use the list of values to select a different admit term. To implement this type of scenario, use the List of Values Setup component to assign the constant value of *Fall 2010* for the Admit Term field.
   
   If the Source value is not set up on the List of Values Setup component, skip this step.
5. Look up the Excluded Values on the List of Values Setup component.
6. If no LOV Context passed or no *Default* LOV Context setting exists for the record and the name combination passed, use the prompt table defined inside Application Designer for the record and the field passed inside the LOV request message.
7. Prepare the response.

Output: |
---|
- After completing the processing steps, the service operation returns a list of values for the user to select. The operation will have no output if an entry does not exist for the requested list of values.
Error Conditions: The following condition results in a service error:

- The online application has not supplied the required record name and field name input parameters.
- The online application has not supplied the required prompt key input parameters.
- The online application has supplied invalid record name or field name.

**Input**

**Message:** SCC_LOV_REQ

The following diagram shows the input parameters that the SCC_GET_LOV service operation receives from an online application:

![Figure 21: SCC_LOV_REQ Message Parameters](image)

The following input parameters are mandatory for the online application to pass to the service operation:

- Field Name
- Record Name
- LOV Context (optional)

The following is an example of the SCC_LOV_REQ message that the SCC_GET_LOV service operation receives from an online application:

```xml
<?xml version="1.0"?>
<SCC_LOV_REQ xmlns="http://xmlns www.w3.org/2001/XMLSchema">
  <LOVS>
    <LOV name="ADM_APPL_DATA_ACAD_CAREER">
      <FIELDNAME>ACAD_CAREER</FIELDNAME>
      <RECORDNAME>ADM_APPL_DATA</RECORDNAME>
      <LOVCONTEXT></LOVCONTEXT>
      <KEYS>
        <KEY>
          <FIELDNAME>INSTITUTION</FIELDNAME>
          <FIELDVALUE>PSUNV</FIELDVALUE>
        </KEY>
      </KEYS>
    </LOV>
  </LOVS>
</SCC_LOV_REQ>
```
Output

Message: SCC_LOV_RESP

When the Integration Broker receives the SCC_LOV_REQ message, it responds with the SCC_LOV_RESP message.

The following diagram shows the output parameter that the SCC_GET_LOV service operation passes to the online application:

Figure 22: SCC_LOV_RESP Message Parameters

The following is an example of the SCC_LOV_RESP message that the SCC_GET_LOV service operation transmits to the online application:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<SCC_LOV_RESP
 xmlns="http://xmlns.oracle.com/Enterprise/HCM/services">
<LOVS>
  <LOV name="ADM_APPL_DATA_ACAD_CAREER">
    <VALUES>
      <VALUE>
        <CODE>BUSN</CODE>
        <DESC>Graduate Business</DESC>
      </VALUE>
      <VALUE>
        <CODE>GRAD</CODE>
        <DESC>Graduate</DESC>
      </VALUE>
      <VALUE>
        <CODE>LAW</CODE>
        <DESC>Law</DESC>
      </VALUE>
      <VALUE>
        <CODE>MEDS</CODE>
        <DESC>Medical School</DESC>
      </VALUE>
      <VALUE>
        <CODE>TECH</CODE>
        <DESC>Technical</DESC>
      </VALUE>
      <VALUE>
        <CODE>UENG</CODE>
        <DESC>Undergraduate Engineering</DESC>
      </VALUE>
      <VALUE>
        <CODE>UGRD</CODE>
        <DESC>Undergraduate</DESC>
      </VALUE>
    </VALUES>
  </LOV>
</LOVS>
</SCC_LOV_RESP>
```
**Fault:** SCC_LOV_FAULT

Refer to the SAD_FAULT_RESP message example in the CREATE Application section.
Configuration Hints and Tips

This section provides post installation setup recommendations for AAWS and the additional three web service operations. For information about the configuration options on the Campus Solutions pages, refer to the Recruiting and Admissions PeopleBooks updates for the Feature Pack 3 release.

For the Integration Broker configuration options, refer to the AAWS Developer's Guide.

Constituent Transaction Manager

Navigation: Set Up SACR > System Administration > Utilities > Constituent Transaction Mgmt > Transaction Setup

Two sample transactions are delivered with AAWS:

1. ADMISSIONS_APPLICATION
2. ADMISSIONS_REGISTRATION

Recommendations:

- Configure Search Match on the Search/Match Setup tab. By default, the system selects the Real Time option for the delivered ADMISSIONS_APPLICATION transactions. You can change the Search Match setting to Batch for this transaction if preferred.
- Create new transactions if you require specific Search Match rules or data update rules for different types of applications (for example, GRAD versus UGRD applications). Map each transaction with a career on the Appl Config page (Application Configuration component).
- Do not mix the Partitioning options for your transactions. If you choose to partition by Transaction, do so for all transactions. If your choice is partition by constituent, the same applies. The exception to this is New User Registration. Partitioning it ignored for the New User Registration transaction.
- It is anticipated that most institutions will prefer that the application is submitted prior to running search/match for an unknown applicant (EMPLID is not known) if the Realtime setting is selected. Be aware that if you select to run Search/Match realtime on Save that the applicant will need to have entered sufficient information for your preferred search/match rules. It is the very first save of application that will trigger the search/match process if this is your configuration. Any subsequent saves prior to submit will update production constituent data if an EMPLID is assigned dependent on your data update rule settings.

Data Update Rules

Navigation: Set Up SACR > System Administration > Utilities > Constituent Transaction Mgmt > Data Update Rule Entry

Three sample data update rules are delivered with AAWS:

1. DEFAULT_UPDATE_RULE
2. ADD_IF_BLANK_RULE
3. DO_NOT_UPDATE_RULE

These serve as examples of how an institution-built online application can access the production data.
Note that the data update rule is applicable only for actions performed by your online application and not for the actions that an administrative user performs using the Campus Solutions pages or components. For example, data update rule configurations does not impact the External Test Score Load application engine process.

The following constituent entities are not included in the Data Update Rules due to their key structures.

- **Relationships and child records of Relationships**
  - Keys: EMPLID, RELATIONSHIP_NBR and EFFDT
  - A new record is always inserted.
  - Exception: Relationship Type = Spouse
    - Only one spouse record is allowed.
    - A new effective dated row is inserted for this relationship type.
  - NOTE: Creation of reciprocal relationships is not included in AAWS.

- **Publications**
  - Keys: EMPLID and PUBLICATION_NBR
  - A new record is always inserted.

- **Work Experience**
  - Keys: EMPLID and SEQUENCE_NBR
  - A new record is always inserted.

- **Honors and Awards**
  - Keys: EMPLID and DT_RECVD
  - A new record is always inserted.

It should also be noted that supporting admissions application data is not included as part of the data update rules. To be consistent with existing processes that update this data, the rules are as follows for the following areas:

- **Academic Interests**
  - If data exists on ADM_INTRST_HDR for an existing EMPLID and ACAD_CAREER and the max ADM_INTRST_HDR.EFFDT is not equal to %Date, a new row is inserted with EFFDT = %Date. Rows are inserted into ADM_INTERESTS for each academic interest populated on SCC_STG_ADMINT. PRIORITY is populated based on ADM_INTRST_HDR.USE_PRIORITY.
  - If data exists on ADM_INTRST_HDR for an existing EMPLID and ACAD_CAREER and the max ADM_INTRST_HDR.EFFDT is equal to %Date, a row is not inserted into ADM_INTRST_HDR. Rows will be inserted into ADM_INTERESTS only where SCC-STG_ADMINT.EXT_SUBJECT_AREA does not exist on ADM_INTERESTS for the current effective date. If ADM_INTRST_HDR.USE_PRIORITY is ‘Y’, the max ADM_INTEREST.PRIORITY is determined and incremented to the next number when new rows are inserted.

- **External Education**
  - A new Data Number (EXT_DATA_NBR) is inserted to EXT_ACAD_DATA if the EXT_ORG_ID on the staging record SCC_STG_ACADHST exists on the core record ACAD_HISTORY. The max EXT_DATA_NBR existing on EXT_ACAD_DATA is determined and incremented to the next number on insert.
  - All child records are populated with the same EXT_DATA_NBR that is inserted on EXT_ACAD_DATA if the field exists on the child record.
  - Exception: For the record EXT_DEGREE, if data exists on EXT_DEGREE where DESCR, DEGREE_DT, DEGREE_STATUS and LS_DATA_SOURCE = data from staging record (SCC_STG_EXTDEGR), a new row is not inserted.

- **General Materials**
  - A new record is always inserted.
  - The max GENL_MATL_NBR is determined and new incremented to the next number when new rows are inserted.

- **Residency Official and Residency Appeals**
  - If the keys (INSTITUTION, ACAD_CAREER and EFFECTIVE_TERM) do not exist on RESIDENCY_OFF or RESIDENCY_APEAL, a new row is inserted.
  - If the keys exist, a new row is not inserted and the data is not updated.

- **Self Reported Residency**
  - If Date Reported (SCC_STG_RES_SLF.SELF_REPORT_DT) does not exists on RESIDENCY_SELF, a new row is inserted.
If Date Reported (SCC_STG_RES_SLF.SELF_REPORT_DT) exists, a new row is not inserted and the data is not updated.

- **Test Scores**
  - If SCC_STG_TEST.TEST_ID does not exist on STDNT_TEST for the EMPLID, a new row is inserted along with any child rows (STDNT_TEST_COMP).
  - If the following elements on STDNT_TEST_COMP match the elements on the staging record SCC_STG_TEST_CMP, update Score, Percentile and Date Loaded regardless of the previous score reported: EMPLID, TEST_ID, TEST_COMP, TEST_DT and DATA_SOURCE.
  - If there is not an existing row on STDNT_TEST_COMP that matches on EMPLID, TEST_ID, TEST_COMP, TEST_DT, DATA_SOURCE, a new row is inserted.

**Recommendations:**

- Use one of the delivered rules as a basis to create your institution specific update rule(s) by clicking the **Copy** button.
- Associate your new update rule with a transaction on Transaction Setup page. Data update rules do not have any impact on User Registration and other such transactions where you do not move the data from staging to production tables. You must define the data update rules for any new entity that has been added through the Entity Registry page where the Apply Data Update Rule checkbox is selected. Additional coding is also necessary to incorporate a new entity into the data update rules.
- For those fields or entities that you are not using as part of your transaction, set the Update Action to **Do Not Update** so you don’t unintentionally blank or zero out data that may be populated by other business processes.
- In the case of effective dated entity data such as Biographical History: If data update rules allow the data in the Campus Solutions system to be updated but the incoming data is same as the data already in the system, then a new EFFDT row is *Not* created (no data update occurs). Suppose, you have set up a rule that gender can be updated by an online application. The system has the latest biographical history record, effective dated April 26, 2009, with information such as Gender as Female and Marital Status as Single. The applicant on April 27, 2009 submits the same biographical history information such as Gender as Female and Marital Status as Single. In this case, because the information is the same, the system does not enter a new effective-dated row. The latest effective-dated row remains the same in the PeopleSoft system, that is April 26, 2009.
- For the Update Action **Add if Blank**, an existing Gender of *Unknown* is considered to be blank and will be updated. As well, if the National ID is populated with the Country Default (999-99-9999 for example), it is considered to be blank and will be updated.
- Keep in mind the relationship of a field to another field when defining your update actions for individual fields in an entity. For example, do not set State/Province to **Update/Insert** if you have Country set to **Do Not Update**. If the system already has a Country value of *Canada* and a Province value of *Alberta*, but the incoming data has a Country value of *Australia* and a Province value of *Tasmania*, you will get an error when posting the data as *Tasmania* is not a valid Province for *Canada*.

If the data update rule is set to **Insert/Update** for an entity or an entity field, the user can use the online application to blank out the data that is maintained in the PeopleSoft system. Assume that you have set up the Marital Status entity field value to **Insert/Update** and for an applicant; the system already has a Martial Status value of *Single*. When the applicant using an online application submits data that includes Marital Status as blank, the Marital Status value in the system becomes blank. In the example above, if you don’t want the Marital Status to become blank, you should select the Update Action of **Safe Update**.

For **Update/Insert** specifically, handling of blanks or zero values in incoming data is as follows:

- For **TYPED** data, being entities with a built in Type (such as Names and Addresses), the blanking out of fields during update is allowed as these entities are atomic in nature and must be updated as a single unit
- For **EFFDT** and non-**EFFDT** entities (which are not typed), we allow the blanking or zero set of entity fields for the **Update/Insert** Data Update Action. For this reason, care should be taken for each entity or field when selecting this option. There may be circumstances where it is desired that incoming data blank out or set a field to zero and other cases where this is not desired.
For **Safe Update** specifically, handling of *blanks* or *zero* values in incoming data is as follows:

- For **TYPED** data, being entities with a built in Type (such as Names and Addresses), the blanking out of fields during update is allowed as these entities are atomic in nature and must be updated as a single unit. For this reason, Safe Update is not shown as an option for entities with a Type.
- For **EFFDT** and **non-EFFDT** entities (which are not typed), we will not blank out or set an existing field to blank or zero if **Safe Update** is selected as the update action. For example, on your transaction, you require Birth Date and Birth Country to be completed but you don’t have Birth Location or Birth State/Province on your transaction. The existing record has values populated for Birth Location and Birth State/Province. By selecting **Safe Update**, the values for Birth Location and Birth State/Province will not be updated to blank when the transaction data is posted.

### Entity Registry

**Navigation:** Set Up SACR > System Administration > Entity > Entity Registry

The entities for all core data records within the constituent and application areas are delivered within the Entity Registry component. The entity data is considered system data and is necessary for the correct functioning of AAWS.

**Recommendations:**

- Register new institution-specific extension entities as children to any entity in the registry. For example, if you want to add an Interaction Mode entity to capture the applicant's preferred social interaction medium, you need to add a new entity Interaction Mode and associate this new entity with an existing parent entity, such as the Constituent entity.
- Read the *PeopleSoft Enterprise Campus Community Fundamentals 9.0 PeopleBook*, “Setting Up Entity Registry”.

### Application Configuration

**Navigation:** Set Up SACR > Product Related > Recruiting And Admissions > Application Configuration > Application Configuration

**Recommendations:**

- Ensure that all academic careers that you will be exposing to your online application user interface are mapped to a valid transaction on the Appl Config tab.
- Ensure that all the academic career, program, and plan combinations that you will be exposing to your online application are mapped to a valid application center on the Application Setup tab.
- If you do not want to create a prospect record when an application is saved in the staging tables or posted to production tables, then no configuration is required on the Prospect Setup tab.
- If prospect record creation is required when an application is saved or posted, then you need to set up values on the Prospect Setup tab.
- On the School Setup tab in the Unlisted School section, the Post Application Using Default option will apply only when an applicant enters a minimum of one unknown school. If multiple unknown schools are entered by the applicant, then the processing defaults to Require Manual Reconciliation.
List of Values

Navigation: Set Up SACR > Product Related > Recruiting And Admissions > Application Configuration > List of Values

Sample configuration is provided for all the commonly used List of Values prompts. You can alter or add the configuration as necessary.

Recommendations:

- List of Values configuration is not mandatory for all prompts. For example, the delivered values for Academic Load include Full-Time and Part-Time. If you do not want to define new values for Academic Load or exclude existing active translate values, you need not use the List of Values configuration.
- You can override the default list of values that is displayed by using the List Of Values Setup component to point to a different database table column or translate field. For example, suppose you have created a table column ACAD_LOAD where you have stored academic load values such as Online and Distance Education. Rather than providing the applicant the translate values of Part-Time and Full-Time, you want the system look at the table column ACAD_LOAD for retrieving the List Of Values for the Academic Load field. In this case, you must use List of Values configuration to override the system behavior of looking at the translate values.
- List of Values configuration supports both online application and Administrative online application pages. For example, if you want to point to a different table column for the Last School Attended field on a power user online application entry page, use the List of Values configuration.
- In the List of Values Setup page, in the Prompt Table Filters grid, ensure that you have defined all the keys included inside the prompt table selected to retrieve the correct List of Values. For example, for the field EXT_ORG_ID, the prompt table key field EXT_ORG_TYPE is set to School to ensure that only external organization IDs that are schools are displayed as the List of Values for the Last School Attended field.
Appendix

This appendix discusses:

- WSDL diagrams.
- Sample code for an online application.

WSDL

Three wsdl's are generated during setup used by AAWS, one for each service

- SAD_ADMISSIONS
- SCC_USERREG
- SCC_LOV

Admissions Application Web Services

The following diagram lists the SAD_ADMISSIONS web service operations and their messages. The diagram also shows an example of a location from where an online application can access the SAD_ADMISSIONS web service.
Figure 23: SAD_ADMISSIONS Web Service and Example of a Locator

User Account Web Service

The following diagram lists the SCC_USERREG web service operations and their messages. The diagram also shows the location from where an online application can access the SCC_USERREG web service.
List Of Values Web Service

The following diagram lists the SCC_LOV web service operations and their messages. The diagram also shows the location from where an online application can access the SCC_LOV web service.

Sample Code

The following example shows how an online application uses Adobe Flex ActionScript™ to call the Application Save web service operation:

Calling SaveApplication – Adobe Flex

```
<mx:XML id="saveApplication" format="e4x">
  <SAD_SAVEAPPL_REQ>
  </SAD_SAVEAPPL_REQ>
</mx:XML>
```
/* Call the Application Save webservice */
public function doSaveApplication():void {

    CursorManager.setBusyCursor();

    WS = new WebService();

    //location of the Web Service Description
    WS.wSDL = "http://"+hostname+"/PSIGW/PeopleSoftServiceListeningConnector/SCC_ADMISSIONS.1.wsdl";

    // Add WS-Security headers if we currently authenticated (logged on)
    toWSSec(WS);

    //and we want the results returned in the e4x XML format
    WS.SAD_SAVEAPPL.resultFormat = "e4x";

    //specify the method that will handle the results
    //returned by the method
    WS.SAD_SAVEAPPL.addEventListener("result", saveApplicationHandler);

    //specify the method that will handle any faults
    WS.SAD_SAVEAPPL.addEventListener("fault", faultHandler);

    //specify the method that will handle the
    //the event of loading the WSDL
    //once the WSDL is loaded we will call
    //our method in the function loadHandler
    WS.addEventListener("load", loadHandler);

    //Load the WSDL for this WebService
    WS.loadWSDL();

    //Construct the Application request
    var req:XML = saveApplication.setChildren(myAppl);

    //Collect the updated constituent information
    var conNS:Namespace = new Namespace(constituent.namespace());

    constituent.conNS::PER_NAMES.conNS::PER_NAME.conNS::NAME_TITLE.setChildren(NameTitle.text);

    constituent.conNS::PER_NAMES.conNS::PER_NAME.conNS::FIRST_NAME.setChildren(NameFirst.text);

    constituent.conNS::PER_NAMES.conNS::PER_NAME.conNS::LAST_NAME.setChildren(NameLast.text);

    constituent.conNS::EMAIL_ADDRESSES.conNS::EMAIL_ADDRESS.conNS::EMAIL_ADDR.setChildren(EmailHome.text);

    constituent.conNS::ADDRESSES.conNS::ADDRESS.conNS::ADDRESS_1.setChildren(AddressHome.text);

    constituent.conNS::BIRTHDATE.setChildren(dateFormatter.format(BirthDate.text));

    constituent.conNS::PERSON_DATA_EFFDTS.conNS::PERS_DATA_EFFDT.conNS::SEX.setChildren(SEX.text);

    req.appendChild(constituent.conNS::PER_NAMES.conNS::PER_NAME.conNS::NAME_TITLE);
    req.appendChild(constituent.conNS::PER_NAMES.conNS::PER_NAME.conNS::FIRST_NAME);
    req.appendChild(constituent.conNS::PER_NAMES.conNS::PER_NAME.conNS::LAST_NAME);
    req.appendChild(constituent.conNS::EMAIL_ADDRESSES.conNS::EMAIL_ADDRESS.conNS::EMAIL_ADDR);
    req.appendChild(constituent.conNS::ADDRESSES.conNS::ADDRESS.conNS::ADDRESS_1);
    req.appendChild(constituent.conNS::BIRTHDATE);
    req.appendChild(constituent.conNS::PERSON_DATA_EFFDTS.conNS::PERS_DATA_EFFDT.conNS::SEX);

    WS.SAD_SAVEAPPL.sendRequest(req);
tChildren(genderCombo.selectedLabel);

SSNSegment.SCC_TEMP_ID.setChildren(constituent.conNS::TEMPID.toString());
   constituent.conNS::PERS_NIDS.setChildren(SSNSegment);

   req = req.appendChild(constituent);

   WS.SAD_SAVEAPPL.request = req;
   WS.SAD_SAVEAPPL();
}