



my|CalPERS

**Guide to File Readiness
For
Direct Authorization
Vendors**

Version History

Document Version	Document Revision Description
1.0	Initial release of document
2.0	<ul style="list-style-type: none"> • Added Version History page • Revised footer • Added reference to Schema Documentation now in Technical Toolkit (sec 2.1) • Updated Sections 4.4 and 4.5
3.0	<ul style="list-style-type: none"> • Updated the Interface ID table in sec 3.3.1 and Appendix A
4.0	<ul style="list-style-type: none"> • Removed Section 4 - content will be covered in new document containing Business Partner Readiness testing instructions

Table of Contents

1. PURPOSE.....	1
2. WALKTHROUGH OF INTERFACE DEVELOPMENT PROCESS	2
2.1. Using Available Interface Documentation	2
2.2. Overview of Data and Process Changes	2
2.3. Interface Style Options	3
3. TIPS FOR INTERFACE DEVELOPERS.....	4
3.1. Introduction	4
3.2. XML and XSD Background.....	4
3.3. Development Guidelines	5
3.3.1. File Header Information	5
3.3.2. Order of Fields	5
3.3.3. Required Fields.....	6
3.3.4. No Data To Include In A Data Element	7
3.3.5. Schema Location Data Attribute	8
3.3.6. XSD Sequence	9
3.3.7. XSD Max Length Value	9
3.4. XML File Guidelines	10
4. PREPARING FOR DEPLOYMENT TO PRODUCTION.....	15
5. HOW TO GET HELP.....	16
APPENDIX A - OVERVIEW OF GUIDES TO FILE READINESS.....	17
APPENDIX B - RESPONSIBILITIES BY PHASE.....	19
APPENDIX C – GUIDE TO XML STRUCTURE	21

1. PURPOSE

The purpose of this Guide to File Readiness is to inform CalPERS business partners about the steps required to create and test an interface file (where elected) to exchange business data with the new CalPERS business system (my|CalPERS) when it goes live.

This guide describes the interface development and testing processes for business partners who intend to report or receive the following data via file to my|CalPERS:

- Deduction requests to add a new deduction
- Deduction requests to change or cancel an existing deduction
- Deduction Register

Some business partners will be creating one interface file to submit deduction requests or receive the Deduction Register and others will be preparing and testing both interface files. While the content herein is applicable to both interface files, the guide calls out, where appropriate, those steps that apply only to a specific interface file. Some of the screen illustrations shown are relative to payroll processing. These examples are meant to be illustrative of the type and format of the functionality contained in all interface processing.

In addition, some Direct Authorization Vendors may also interact with CalPERS as an employer, a health carrier, or a dental carrier. In order to assist CalPERS' business partners with their interactions, three separate Guides to File Readiness have been developed. If your organization falls into multiple categories, it is recommended that you review each of the appropriate guides. Information regarding all the guides is included in Appendix A, [Overview of Guides to File Readiness](#).

Lastly, a general overview of the responsibilities of CalPERS or business partner during each of the my|CalPERS implementation phases has been included in Appendix B, [Responsibilities by Phase](#).

2. WALKTHROUGH OF INTERFACE DEVELOPMENT PROCESS

The following sections, 2.1 through 2.4, provide a tutorial for creating interface files for my|CalPERS. The topics within the tutorial include:

- Using Available Interface Documentation
- Overview of Data and Process Changes
- Interface Style Options
- Tips for Interface Developers

2.1. USING AVAILABLE INTERFACE DOCUMENTATION

Each interface can be seen as the data connection between business systems. The technical documentation used to define an extract file's format and the mechanism used to transport that file to another party is referred to as an interface specification. The specifications for the deduction request and Deduction Register interfaces for my|CalPERS are available in the [Technical Toolkit for Direct Authorization Vendors](#) in the Business Partner area of CalPERS On-Line.

The toolkit is comprised of several documents in a WinZip file that can be easily downloaded and updated periodically. The toolkit is intended to help clarify the application of the interface specifications for business partners to facilitate successful creation and test of an interface file. The toolkit documentation includes but not limited to:

- Data Element Definitions
- XML File Structures
- Sample XML Files
- XSD Schemas
- Validation Rules
- CalPERS ID File Exchange Layout
- FTP Encryption/Decryption Requirements

Please take time to become familiar with all the artifacts located in the toolkit paying special attention to the [DAV's Guide to the Technical Toolkit](#) and the [Reporting Deduction Information to my|CalPERS](#) and [Deduction Registers](#) to understand how the components of the interface specifications fit together and are intended to be used.

2.2. OVERVIEW OF DATA AND PROCESS CHANGES

With my|CalPERS, business partners will have the ability to transmit data files via file upload or File Transfer Protocol (FTP). The process to submit files by upload in my|CalPERS will be new and different than any data transmission option used today. The opportunity for employers to

exchange data with CalPERS through FTP exists now and will continue with the my|CalPERS system implementation.

In addition to file upload as a new reporting method, some key data and process changes associated with my|CalPERS deployment include:

- Electronic data reporting only; no tapes, diskettes or paper documents
- Files will be formatted according to Extensible Mark-up Language (XML)
- Business partner responsibility for error correction by:
 - Using on-line system screens, or
 - Correcting invalid records in native system and submitting a new file
- Files contain new data elements including system generated CalPERS IDs to identify business partners and participants

The best source for reviewing the new data elements is the [Technical Resources](#) documents located in the Business Partner area of CalPERS On-Line.

2.3. INTERFACE STYLE OPTIONS

CalPERS is offering two ways to submit interface files for business partner transactions. Both styles are available for submitting deduction requests. The styles are:

1. File Upload (HTTPS) – The business partner uploads a file through my|CalPERS application functionality. This style requires user interaction with the application to submit the file.
2. Secured File Transfer Protocol (SFTP) – The business partner submits the files using encryption and SFTP. Unlike the upload process, this style can be fully automated. Additionally, my|CalPERS will provide a response file that indicates errors that can be processed for correction within the business partner's system.

The Deduction Register can be received via SFTP or downloaded as a Comma Separated Values (.csv) file using the online system.

3. TIPS FOR INTERFACE DEVELOPERS

3.1. INTRODUCTION

This section is included to assist CalPERS business partner's technical staff with some of the nuances associated with developing the XML file as specified by the new my|CalPERS system. The my|CalPERS solution incorporates a two-step file validation concept.

The first file validation is referred to as File Level 1 and this Level 1 validation is looking to enforce the XML interface file specification or XML Schema Document (XSD) against the file that is being submitted. The main function of Level 1 validation is to make sure that all required fields are present and the file being submitted is a proper XML file, meaning that start tags and end tags are present, no empty tags are submitted and maximum length of a data element is enforced. This level also checks each file submitted, via File Upload and FTP, to ensure the proper file naming convention is being used.

File Level 2 validation consists of enforcing the defined CalPERS business rules against the data contained in the XML file. Essentially, Level 1 is checking that the file can be consumed by my|CalPERS and Level 2 is checking the actual data contained in the XML file. This section is going to focus on the Level 1 validation nuances since Level 2 errors are well described by the error messages. For SFTP file submissions, Level 2 errors will be returned through a response file that will identify the errors. These can then be corrected in the business partner's own system and the records can be resubmitted. Alternatively, Level 2 errors can be displayed on a user screen within my|CalPERS where they can be corrected directly.

This section is meant to serve as a supplement to XML coding standards and principles that are readily available on the Internet. Information on various XML tools and resources is also available at the links identified in section 2.1, [Using Available Interface Documentation](#), and section 5, [How to Get Help](#), of this document.

3.2. XML AND XSD BACKGROUND

CalPERS is in the process of moving all file reporting to an XML format from a flat fixed length text format. This move to XML was decided in large part to allow data transfer or document exchange within or across organizations with different platforms. In order to establish a jumping off point, XSD's must also be discussed. XSD defines syntax and shows how elements and attributes in an XML file should be contained.

Some of the XSD's that CalPERS is using for the my|CalPERS solution have some nuances and idiosyncrasies that this guide will clarify and discuss so as to give the developer a better understanding of how to interact with CalPERS' new system. Those nuances are discussed in the next section 2.4.3, [Development Guidelines](#).

It is the business partner's responsibility to prepare a compliant file according to CalPERS specification. XML format allows files to be validated against the XSD schemas that the files are designed to comply with. Tools are available to assist business partners with file validation against the XSD schemas provided in CalPERS specifications.

3.3. DEVELOPMENT GUIDELINES

This section of the guide will review some of the more common problem areas and common questions when developing a CalPERS-defined XML file as well as what may be the best way to handle those problems. A Guide to XML Structure can be found in [Appendix C](#), at the end of this document.

3.3.1. FILE HEADER INFORMATION

Every file has an element defined as Interface File Type. The Interface File Type is specific to the internal CalPERS-defined interface specifications. Table 1 shows the Interface File Type values related to each file that is being submitted or received. The Interface Number value on the inbound file should be used for the Interface File Type. The deduction request example below shows the data element and the value when submitting an XML file for processing:

```
<cuns:InterfaceTypeId>20016</cuns:InterfaceTypeId>
```

Table 1- Interface File Type

GUIDE	INTERFACE	INTERFACE DESCRIPTION	INTERFACE DIRECTION	INTERFACE NUMBER
Direct Authorization Vendors	Submit Direct Authorization Information	Used to process direct authorization requests from business partners	Inbound to my CalPERS	20016
	Receive Deduction Registers for Vendors	Used to send applicable parties information about the monthly deduction taken from retiree benefits on their behalf	Outbound from my CalPERS	20010

3.3.2. ORDER OF FIELDS

There is a set order (hierarchy) for the XML contained in the interface files. The data elements must be represented in a certain order. The hierarchy for each interface file is defined in its associated XSD; therefore the XSD is the driving set of specifications when developing the XML file. There are two sets of examples below demonstrating the hierarchy of data. The first example shows the correct order in which the data elements should be represented. The data element in bold in the second example is not in the correct order. This would cause the XML file to be rejected because the data elements are not in the correct order as per the XSD.

Example 1 (Deduction Request) –data elements in correct order

```
<n1:DeductionDataDetail>
  <n1:TransactionCode>ADD</n1:TransactionCode>
  <n1:AgreementId>123456789</n1:AgreementId>
```



```
<n1:DeductionAmount>50.00</n1:DeductionAmount>
<n1:DeductionFrequency>REG</n1:DeductionFrequency>
</n1:DeductionDataDetail>
```

Example 2 (Deduction Request) – data elements not in correct order

```
<n1:DeductionDataDetail>
  <n1:TransactionCode>ADD</n1:TransactionCode>
  <n1:DeductionAmount>50.00</n1:DeductionAmount>
  <n1:DeductionFrequency>REG</n1:DeductionFrequency>
  <n1:AgreementId>123456789</n1:AgreementId>
</n1:DeductionDataDetail>
```

Also note that there is a start tag and an end tag for each data element. The start tag always opens with a less than sign “<” whereas the end tag always starts with a less than sign followed by a slash “</”. This is shown in the above example.

3.3.3. REQUIRED FIELDS

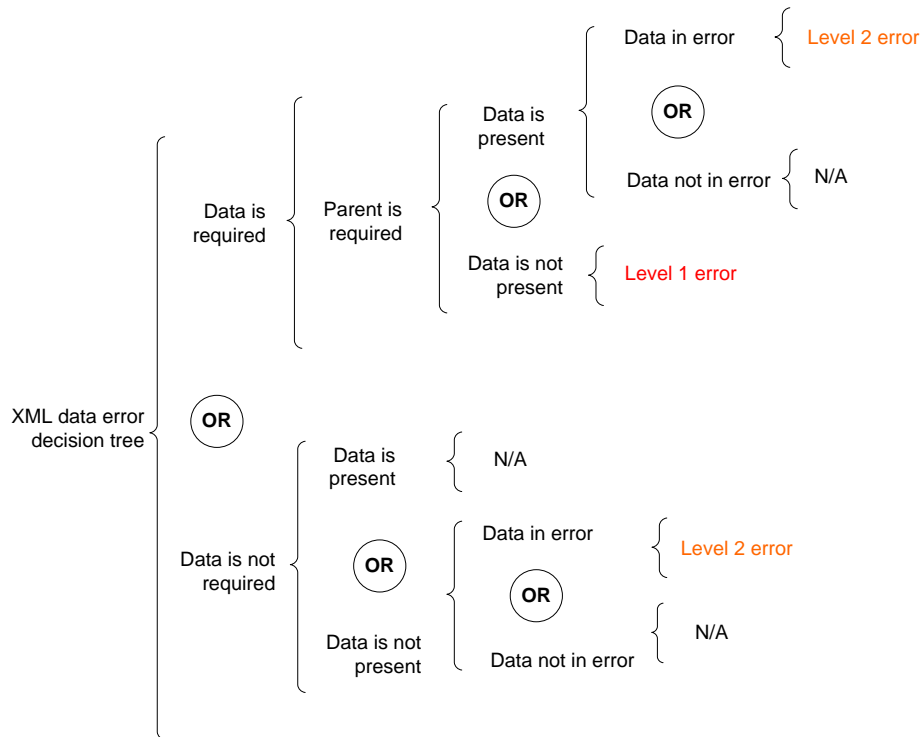
If a field is defined as required and its parent element is also required (or it has no parent element), then it must always be present in the file. Otherwise the file will fail Level 1 validation.

For example, in the Deduction Request file, the data elements <ParticipantIdType> and <ParticipantId> are required fields and must always be included in the XML file:

```
<n1:ParticipantIdType>PID</n1:ParticipantIdType>

<n1:ParticipantId>1223344565</n1:ParticipantId>
```

The following diagram depicts how Level 1 and Level 2 errors may be generated in my|CalPERS when required fields are missing in the XML files.



3.3.4. NO DATA TO INCLUDE IN A DATA ELEMENT

When using XML, if a data element is not going to contain data and it is not a required or conditional field then the tags must be left out. If a tag is included and there is no data for the data element then the XML file may fail Level 1 Validation. Using the deduction request XSD as an example, if a new deduction request is submitted with `<ParticipantIdType>SSN</ParticipantIdType>`, the `<ParticipantSSN-4>` element tags are not to be included. The example below shows how a new deduction request would be reported in the XML file. From this example it can be seen that no Participant SSN-4 data element exists.

```

<n1:ParticipantDeductions>
  <n1:ParticipantIdType >PID</n1:ParticipantIdType>
  <n1:ParticipantId>1122334455</n1:ParticipantId>
</n1:ParticipantDeductions>
  
```

3.3.5. SCHEMA LOCATION DATA ATTRIBUTE

The XML schema language defines the SchemaLocation attribute to provide information to the software that processes the XML file as to where the XSD may be found. This attribute uses pairs of values where the first URL reference in each pair is a namespace name, and the second is the location of a schema that describes that namespace. Since the SchemaLocation attribute is a location pointer, it is important to make sure that a space is entered between the first and second namespace. The following deduction request example shows a portion of the XML file with the SchemaLocation attribute.

Example of SchemaLocation Attribute:

```
<soap:Body>  
<n1:DADeductionRequest  
  xsi:schemaLocation="http://calpers.ca.gov/PSR/DADeductionRequestV1  
  DADeductionRequestV1.xsd"
```

3.3.6. XSD SEQUENCE

The XSD defines the sequence in which data elements are to be included in the XML file. When an <xs:sequence> entry is shown in the XSD the XSD will then follow with the order in which the elements need to appear in the XML file. If the order is changed, the XML file will fail Level 1 validation. In the deduction request example below if the order is changed, for example, and <AgreementId> comes before <TransactionCode>, the file will fail validation.

```
<xs:sequence>
  <xs:element name="TransactionCode" type="cuns:CodeType" />
  <xs:element name="AgreementId" type="cuns:IdType" minOccurs="0"/>
  <xs:element name="DeductionAmount" type="xs:Decimal" />
  <xs:element name="DeductionFrequency" type="cuns:CodeType" />
  <xs:element name="DeductionStartDate" type="cuns:LocalDateType" minOccurs="0" />
    <xs:annotation>
      <xs:documentation>Conditional:
1. Required if Deduction Type is UCRS and Deduction frequency is one-time or limited Term.
2. Required if Deduction Type is CalPERS Long Term Care.
    </xs:documentation>
    </xs:annotation>
  </xs:element>
  <xs:element name="DeductionEndDate" type="cuns:LocalDateType" minOccurs="0">
    <xs:annotation>
      <xs:documentation>Conditional:
1. Required if Deduction Type is UCRS and Deduction frequency is one-time or limited term.
2. Required if Deduction Type is CalPERS Long Term Care.
    </xs:documentation>
    </xs:annotation>
  </xs:element>
</xs:sequence>
```

3.3.7. XSD MAX LENGTH VALUE

The Max Length value specification defines the maximum number of characters or digits that are allowed in the data field. This Max Length definition does not mean that the data element must always be the specified length but rather that a value can not be more than the defined maximum length. If the XML Schema specifies a Max Length of three (3) but the code value(s) are two characters in length then the two character value without any padding should be used. For example, if a data element "MaxLengthExample" is specified as having a Maximum Length of 3 but a valid code value for this element is "AA" this would be represented in the XML File as:

```
<cuns:MaxLengthExample>AA</cuns:MaxLengthExample>
```

There are some data elements that must always be a specified length. For example, the SSN-type data elements in the XSD's must always have a length of nine characters. This is indicated by the `xs:length` specification in the Common Utilities XSD:

```
<xs:simpleType name="SSNType">
  <xs:restriction base="xs:string">
    <xs:length value="9"/>
  </xs:restriction>
</xs:simpleType>
```

Since some SSN's have leading zeroes, when this value is added to the XML file it should contain the leading zeroes.

```
<cuns:SSN>001234567</cuns:SSN>
```

Because the SSN must always be a nine digit value, this value may be stored as a string in the source system or, if stored as a numeric value, should always be zero padded to ensure that it will be nine digits in length. In the example above, the SSN would be stored as "001234567" or there should be logic in place to take the numeric value of 1234567 and pad it with zeros to become the nine digit length.

3.4. XML FILE GUIDELINES

In XML, a well-formed file must conform to the following rules, among others:

- **XML Files must have a Root Element**

XML files must contain one element that is the parent of all other elements. This element is called the root element.

```
<root>
  <child>
    <subchild>.....</subchild>
  </child>
</root>
```

Example

```
<n1:DeductionDataDetail>
  <TransactionCode>
    <cuns:CodeType>PID</cuns:CodeType>
  </TransactionCode>
  <AgreementId>
    <cuns:IdType>112233445</cuns:IdType>
  </AgreementId>
  <DeductionAmount>50.00</DeductionAmount>
  <DeductionFrequency>
    <cuns:CodeType>REG</cuns:CodeType>
  </DeductionFrequency>
</n1:DeductionDataDetail>
```

In the example above <n1:DeductionDataDetail> is the root element, <TransactionCode>, <AgreementId>, <DeductionAmount> and <DeductionFrequency> are child elements while <cuns:CodeType> and <cuns:IdType> are sub child elements.

- **All XML Elements must have a Closing Tag**

In XML, it is illegal to omit the closing tag. All elements must have a closing tag:

<p>This is a paragraph</p>

Although in XML, empty elements may be marked with an empty-element (self-closing) tag (such as <Gender/>), in my|CalPERS, empty, or null, data elements should be removed since null values are not processed. Only data elements with values should be included in the files.

- **XML Tags are case sensitive**

XML elements are defined using XML tags. Tags are case sensitive. With XML, the tag <BusinessPartnerCalPERSId> is different from the tag <BusinessPartnercalPERSId>.

Opening and closing tags must be written with the same case:

<TransactionCode>ADD</TransactionCode> Correct!

<TransactionCode>ADD</Transactioncode> Incorrect!

Note: Opening and closing tags are often referred to as start and end tags. Use whatever is preferred. It is exactly the same thing.

- **XML Elements must be properly nested**

Tags may be nested but must not overlap. Each non-root element must be completely contained in another element. In XML, all elements must be properly nested within each other:

```
<n1:ParticipantDeductions>
  <n1:ParticipantIdType>PID</n1:ParticipantIdType>
  <n1:ParticipantId>1122334455</n1:ParticipantId>
</n1:ParticipantDeductions>
```

In the example above, properly nested simply means that since the <ParticipantIdType> element is opened inside the <ParticipantDeductions> element, it must be closed inside the <ParticipantDeductions> element.

```
<TransactionCode>
  <cuns:CodeType>ADD</cuns:CodeType>
</TransactionCode>
```

In the example above, properly nested simply means that since the <cuns:CodeType> element is opened inside the <TransactionCode> element, it must be closed inside the <TransactionCode> element.

- **XML Attribute Values Must be Quoted**

XML elements can have attributes in name/value pairs just like in HTML.

In XML the attribute value must always be quoted. The first example below is correct, the second is incorrect:

Example #1

```
<note date="12/11/2007">
  <to>Tove</to>
  <from>Jani</from>
</note>
```

Example #2

```
<note date=12/11/2007>
  <to>Tove</to>
  <from>Jani</from>
</note>
```

The error in the second example is that the date attribute in the note element is not quoted.

- **Entity References**

Some characters have a special meaning in XML. If a character like < is placed inside an XML element, it will generate an error because the parser interprets it as the start of a new element; this will generate an XML error:

```
<message>if salary < 1000 then</message>
```

To avoid this error, replace the < character with an entity reference:

```
<message>if salary &lt; 1000 then</message>
```

There are 5 predefined entity references in XML:

1. <
<
less than
2. >
>
greater than
3. &
&
ampersand

4. '
,
apostrophe
5. "
"
quotation mark

Note: Only the characters <" and & are strictly illegal in XML. The greater than character is legal, but it is a good habit to replace it.

- **All attribute values are quoted with either single (') or double (") quotes.**

Single quotes close a single quote and double quotes close a double quote. To include a double quote inside an attribute value that is double quoted, or a single quote inside an attribute value that is single quoted, escape the inner quote mark using entity references.

- **Comments in XML**

The syntax for writing comments in XML

<!-- This is a comment -->

- **Never use the two dashes (--) anywhere but at the beginning and end of your comments.**

- **White-space is preserved in XML**
- **With XML, the white-space in a file is not truncated.**
- **Naked ampersand (should be represented as &#38;)**
- **Sequence]]> not allowed in content**

4. PREPARING FOR DEPLOYMENT TO PRODUCTION

As the business partner prepares for production deployment, the following should be considered:

- The my|CalPERS deployment is dependent upon many factors. The normal CalPERS-business partner communication channels will be used to confirm that CalPERS has completed its deployment into production. It would be best for business partners to confirm the my|CalPERS deployment before placing their new systems into production.
- Business partners will have test Deduction Register files in XML and .csv formats available for testing their retrieval and update processes during the testing window.
- Any new deduction requests that occur after the cutoff date can be submitted to my|CalPERS using the online system, File Upload or FTP process after launch. If you are submitting your deductions using Form PRS-346, the cutoff date is August 22, 2011. If you are submitting your deductions using a file, the cutoff date is August 25, 2011. Please understand this processing is for the October 1, 2011 warrants. Please note that after launch, you will be submitting deductions beginning for the next Benefit Roll period, November 1, 2011 warrants. Any payments that are required for benefits provided between the cutoff date and the first post-launch Benefit Roll will need to be collected directly from the participant.
- The primary SAA information and logon data will be migrated from the File Readiness Test environment into the production environment for the initially-identified SAA only. However, all other users established by the SAA during the test period will not be migrated and will have to be recreated.

5. HOW TO GET HELP

CalPERS provides the following resources to address business partner questions:

1. The most up to date information is available at CalPERS Online including [Frequently Asked Questions \(FAQ's\)](#). Just follow the links for [Business Partners](#), [myCalPERS Readiness for Business Partners](#) and [For Direct Authorization Vendors](#) and choose the link for the desired information.
2. The [Technical Resources](#) page in the Business Partner area of CalPERS On-Line includes a document where you will find links to websites that may be helpful in understanding the technologies associated with creating an electronic XML file. These sites include:
 - Products that can be used to analyze an XML file or to convert a flat file to XML format
 - Information to clarify the process for producing an XML document.
3. Business partners can contact the Employer Contact Center (ERCC) which was established to provide employers with a simpler and convenient way to address their business needs. During testing and implementation, this team will expand their support role to include Direct Authorization business partners. Business partners can call one toll-free number **888 CalPERS** (or **888-225-7377**) for help during testing and implementation.
4. To reach a PERT representative, send an email to the PERT Project mailbox at CalPERS_PERT4U@calpers.ca.gov.

APPENDIX A - OVERVIEW OF GUIDES TO FILE READINESS

In order to assist CalPERS' business partners with their interactions, three separate Guides to File Readiness have been developed.

- my|CalPERS Guide to File Readiness for Employers
- my|CalPERS Guide to File Readiness for Direct Authorization Vendors
- my|CalPERS Guide to File Readiness for Health And Dental Carriers and Associations

Some business partners fit into more than one of these categories. For example, a single business partner can be a direct authorization vendor, an association, and an employer. If your organization falls into multiple categories, it is recommended that you review each of the appropriate guides. The following table identifies which interfaces are described in the guides.

Table 2 - my|CalPERS Interfaces

GUIDE	INTERFACE	INTERFACE DESCRIPTION	INTERFACE DIRECTION	INTERFACE NUMBER
Employers	Payroll Contribution	Used to process payroll transactions	Inbound to my CalPERS	10006
	Employer Payroll Report File Download	Used when downloading a payroll report file to correct payroll errors	Outbound from my CalPERS	10049
	Employer Payroll Response	Used to correct payroll errors when file is submitted and returned via FTP	Outbound from my CalPERS	10006
	Retirement Enrollment	Used to process retirement enrollment transactions	Inbound to my CalPERS	00007
	Retirement Enrollment Response	Used to correct health enrollment errors when file is submitted and returned via FTP	Outbound from my CalPERS	00007
	Health Enrollment	Used to process health enrollment transactions	Inbound to my CalPERS	50031
	Health Enrollment Errors Download	Used when downloading a Health Enrollment report file to correct Health Enrollment Errors	Outbound from my CalPERS	50068

GUIDE	INTERFACE	INTERFACE DESCRIPTION	INTERFACE DIRECTION	INTERFACE NUMBER
	Health Enrollment Response	Used to correct health enrollment errors when file is submitted and returned via FTP	Outbound from my CalPERS	50031
Direct Authorization Vendors	Submit Direct Authorization Information	Used to process direct authorization requests from business partners	Inbound to my CalPERS	20016
	Receive Deduction Registers for Vendors	Used to send applicable parties information about the monthly deduction taken from retiree benefits on their behalf	Outbound from my CalPERS	20010

APPENDIX B - RESPONSIBILITIES BY PHASE

The following table describes business partner and CalPERS responsibilities by representative schedule phases. The representative schedule phases for business partners preparing to exchange files with CalPERS include the following:

- Discover
- Design
- Develop
- Test
- Deploy

Phase	Responsibility	Success Criteria
Discover	<p><u>Business Partner:</u></p> <ul style="list-style-type: none"> • Understand the interfaces to be used with CalPERS as well as methods for interacting with CalPERS new system <p><u>CalPERS</u></p> <ul style="list-style-type: none"> • Share information on interfaces to be used by business partner 	<p><u>Business Partner:</u></p> <ul style="list-style-type: none"> • Be able to design internal system changes and extract files for exchange with CalPERS as well as the business processes to interact with CalPERS <p><u>CalPERS</u></p> <ul style="list-style-type: none"> • Business partner understands CalPERS interface designs
Design	<p><u>Business Partner:</u></p> <ul style="list-style-type: none"> • Design system changes to accommodate data for exchanges and to interact successfully with CalPERS <p><u>CalPERS</u></p> <ul style="list-style-type: none"> • Share design information on interfaces to be used by business partner; notify business partner of design changes 	<p><u>Business Partner:</u></p> <ul style="list-style-type: none"> • Complete designs to ensure mandatory data elements for file exchanges will be available under defined conditions in file format specifications <p><u>CalPERS</u></p> <ul style="list-style-type: none"> • Business partner is clear on what data is mandatory under which conditions; design changes are communicated as soon as possible

Phase	Responsibility	Success Criteria
Develop	<p><u>Business Partner:</u></p> <ul style="list-style-type: none"> Implement designed changes to internal systems to allow future file exchanges with CalPERS <p><u>CalPERS</u></p> <ul style="list-style-type: none"> Clarify design questions from business partners 	<p><u>Business Partner:</u></p> <ul style="list-style-type: none"> Changes are implemented and allow the system to operate as intended for exchanging files with CalPERS <p><u>CalPERS</u></p> <ul style="list-style-type: none"> Business partners can complete implementation for valid file exchanges
Test	<p><u>Business Partner:</u></p> <ul style="list-style-type: none"> Test file exchanges with CalPERS to verify operability and error processing requirements <p><u>CalPERS</u></p> <ul style="list-style-type: none"> Provide a test platform and guides for business partners to test their interfaces 	<p><u>Business Partner:</u></p> <ul style="list-style-type: none"> Satisfied that the file exchange is operational and that errors are known and can be corrected <p><u>CalPERS</u></p> <ul style="list-style-type: none"> Provide responses to questions and an operational test platform for use by business partners
Deploy	<p><u>Business Partner:</u></p> <ul style="list-style-type: none"> Deploy system and process changes to operational status when coordinated by CalPERS for production deployment <p><u>CalPERS</u></p> <ul style="list-style-type: none"> Coordinate production deployment timing and conditions with business partners 	<p><u>Business Partner:</u></p> <ul style="list-style-type: none"> Systems and business processes are deployed and operational; file exchanges can continue with CalPERS new system <p><u>CalPERS</u></p> <ul style="list-style-type: none"> Deployment dates and conditions are communicated appropriately to coordinate operational status

APPENDIX C – GUIDE TO XML STRUCTURE

Two XSDs are common across all XML files being exchanged between file reporting business partners and CalPERS:

- CommonUtilitiesV1.xsd
- SoapEnvelope.xsd

The remaining XSD(s) define the body of the XML files. The specific XSDs are called out in the instructions structure guide shown below.

The following describes how the content of the files sent by File Reporting Direct Authorization Vendors and CalPERS must be structured. Files exchanged via file upload or FTP will be structured as a SOAP Envelope.

Envelope

- The interface file must contain a root element named “Envelope” with the namespace identifier of “http://schemas.xmlsoap.org/soap/envelope/”
- An envelope MUST have exactly one child element called soap:Header
- An envelope MUST have exactly one child element called soap:Body
- An envelope MUST NOT have any element children of soap:Envelope following the soap:Body element

Header

- The soap:Header element must have one child element named HeaderInfo with the namespace identifier of “http://calpers.ca.gov/PSR/CommonUtilitiesV1”

Body

- The soap:Body element of a file inbound, to CalPERS, must have one child element named DADeductionRequest with the namespace identifier of “http://calpers.ca.gov/PSR/DADeductionRequestV1”
- The soap:Body element of a file outbound, from CalPERS, must have one child element named DADeductionResponse with the namespace identifier of “http://calpers.ca.gov/PSR/DADeductionResponseV1”

XSD

- Inbound from DAVs to CalPERS – DADeductionRequestV1.xsd
- Outbound from CalPERS to DAVs – DADeductionResponseV1.xsd