Oracle Public Sector Revenue Management

PSRMSS-PSRM Integration Guide

Release 1.1.0.1.0

E53306-02

August 2014

Oracle Public Sector Revenue Management PSRMSS-PSRM Integration Guide

Release 1.1.0.1.0

E53306-02

August 2014

Documentation build: 9.8.2014 15:47:8 [TS_1410216428000]

Contents

Chapter 1

Self Service Integration

Oracle Public Sector Revenue Management ("the product") provides integration with Oracle Public Sector Revenue Management Self Service ("the self service product"). Refer to the self service product documentation for an overview of the implementation for that application. The following sections describe the functionality provided in the product to support the integration and outline the configuration required to support the functionality.

The Big Picture

This section describes high level topics related to the integration.

Message Processing Overview

Oracle Public Sector Revenue Management interacts with the self service product via web services. An appropriate inbound web service is invoked from the self service product via BPEL. The inbound web service is configured with a processing service script that knows how to process the request. Requests fall into one of the following categories:

- A taxpayer request. These types of requests fall into one or more sub-categories:
  - A taxpayer inquiry. If the taxpayer requests information like to find out the status of a refund, the inbound web service references a service script that performs the appropriate logic. Most of the time the service script associated with the inbound service can perform the logic needed to retrieve the information to present to the taxpayer. However, there may be cases where the service script creates a real time Service Task to retrieve the requested information and return it.

    Fastpath: Refer to the *Refund Status* for an example of a request for information that results in the creation of a service task.

  - A request to perform some transaction processing. For example, the taxpayer may submit a payment via self service. In this case, the service script associated with the inbound web service creates a Service Task record that will subsequently perform the necessary validation and processing. The taxpayer does not wait for the service task to complete its lifecycle.

- An internal web service request submitted by the self service product to get information. For example, before submitting a payment the self service product may require the taxpayer to provide proof of identity to verify that the taxpayer is known to the system. The taxpayer identification information is sent to this product via a web service and is processed real time.

Naming convention: The product's inbound web services require that the service name starts with 'TS%'. Specific inbound web services are mentioned throughout the supported functionality.

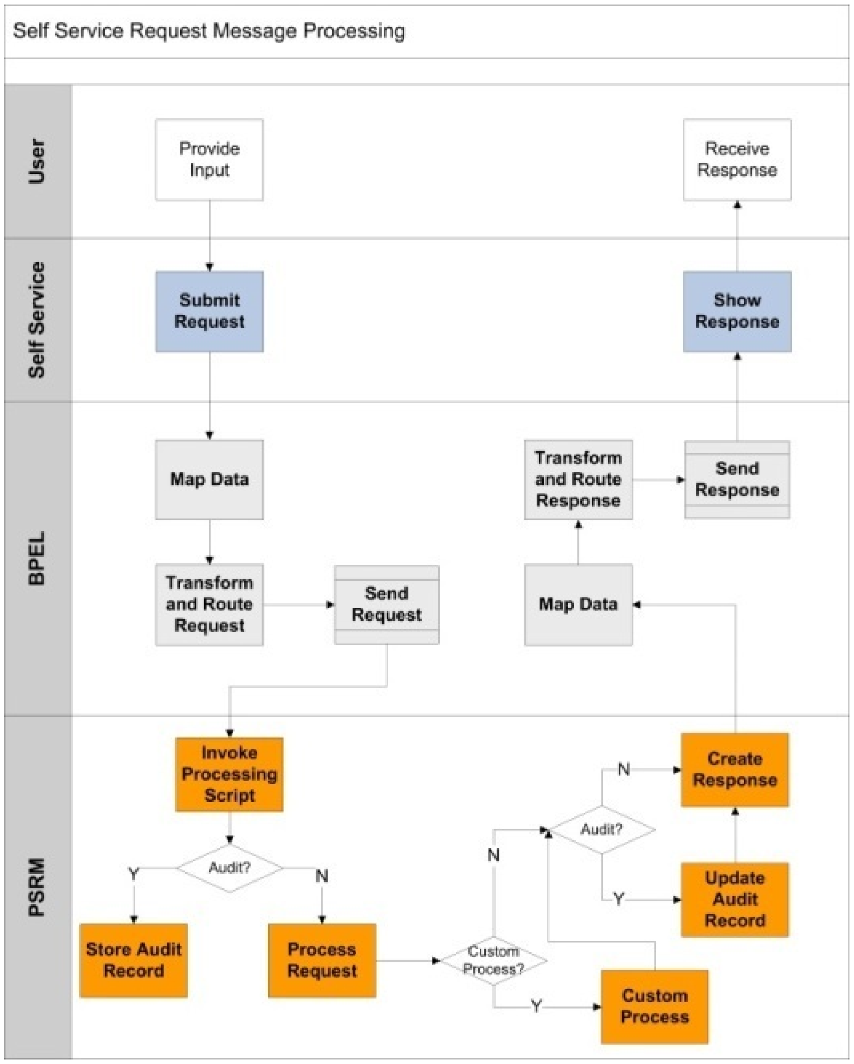


Figure 1: Message Processing

User Identity Verification

The self-service product allows the taxpayer to register as a website user, and enroll to manage taxes online.

A person may work with the self-service portal application in two modes:

- **Casual User.** There is no need to sign in when casually viewing website content or seeking tax-related advice. Certain online services and features are available for casual users.

- **Enrolled User.** A user must log in and be properly authenticated and authorized in order to access sensitive personal and financial information such as historical tax returns, payment history, and other items.

XML request contents are affected as follows:

- Requests initiated by a casual user contain no user or tax account information.
- Requests initiated by an enrolled user contain tax account identifiers.

Note: See the *Tax Account Access Information* topic for a detailed explanation of tax account access.

Some online services include an optional taxpayer identification step:

- For casual users, the identification is performed if it's required by business rules.
- For signed-in users, the identification is not performed.

Tax Account Access Information

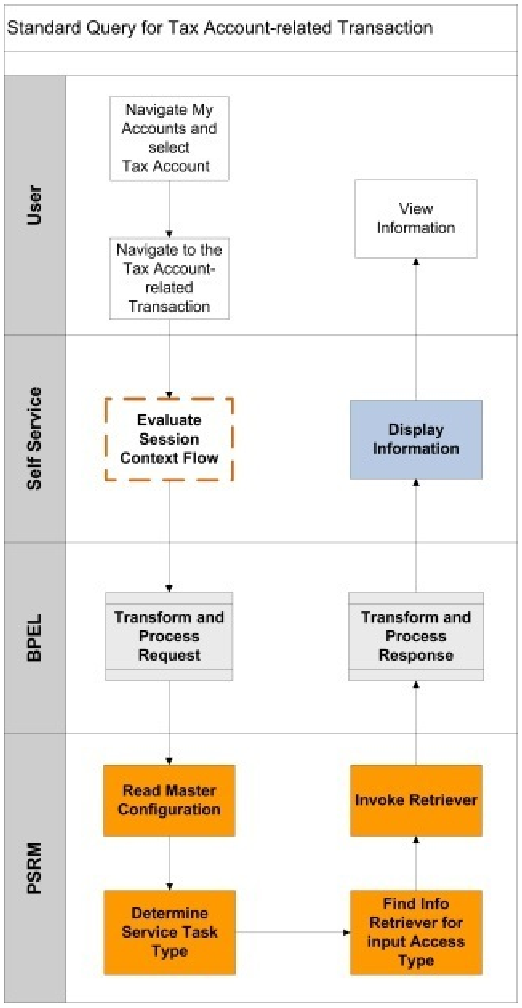
The revenue management authority grants s user access to his/her taxes. The single access “unit” could be a tax role, an account, a tax type, or other entity.

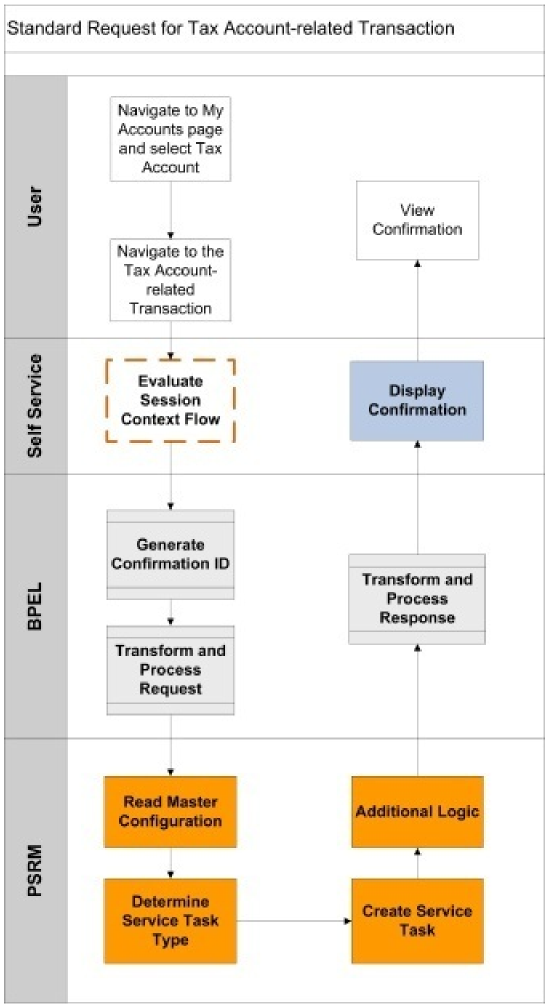
Users can be granted access for many different purposes, including management of a user's tax role, account management, or managing an account for a limited period of time.

When a user is registered and enrolled into online services, the enrollment summary ("tax accounts") is displayed on the user's **My Accounts** portal page. The summary can include tax role(s), accounts, obligations, and other entities.

The request message captures the identifiers of a tax account that is currently being viewed by the self service portal user.

The following diagrams illustrate the process flow for tax account-related inquiry and service requests.


Standard Query for Tax Account-related Transaction
User
Self Service
BPEL
PSRM
Navigate My Accounts and select Tax Account
Navigate to the Tax Account-related Transaction
View Information
Evaluate Session Context Flow
Display Information
Transform and Process Request
Transform and Process Response
Read Master Configuration
Invoke Retriever
Determine Service Task Type
Find Info Retriever for input Access Type


Standard Request for Tax Account-related Transaction
User
Self Service
BPEL
PSRM
Navigate to My Accounts page and select Tax Account
Navigate to the Tax Account-related Transaction
View Confirmation
Evaluate Session Context Flow
Display Confirmation
Generate Confirmation ID
Transform and Process Request
Transform and Process Response
Read Master Configuration
Determine Service Task Type
Create Service Task
Additional Logic

Access Type

Access type describes the scope of a single tax account managed via the self service portal. An example of access type is **Tax Role**.

The access type is referenced in various self service-related configurations; when the request is processed the system uses the access type to determine the appropriate logic to execute. For example, an outstanding balance for a tax role is retrieved in a different way than the outstanding balance for an account.

The <access type> element is included in every message. Supported access types are represented in the product by the lookup **USER_ACCESS_TYPE_FLG**.

Access Keys

The tax account is identified by the access type and a set of keys. The integration supports an ability to identify the account with up to 10 keys.

The first key is always reserved for the taxpayer ID associated with the tax account.

Line of Business

In the web self-service application, Line of Business is used to broadly group areas of interest. The web pages found within a defined line of business are related to the information and tasks appropriate for that area of interest. Some examples of lines of business for an implementation may include Individual taxpayers, Business taxpayers and Tax Preparers. Another implementation may use even more granularity, to include, for example, charitable or non-profit organizations.

In terms of the product, Line of Business can be viewed as a broad category of taxes managed by the tax authority, but also as a way to define a target audience spanning several taxpayer types. Line of business is referenced in self service-related configurations and allows processes to invoke specific logic when needed. For example, different rules may apply when the system identifies tax accounts owned by individuals and businesses during enrollment.

The <lineOfBusiness> element is included in the requests whose processing logic may depend on its value. Supported lines of business are represented in the product by the lookup **LINE_OF_BUSINESS_FLG**.

Confirmation Message

All web services that represent a taxpayer request and result in the creation of a service task within the product are designed to provide the taxpayer with a confirmation ID and a confirmation message. Additional confirmation details may be added by logic in the product that performs actions. This may be used when a taxpayer wants to inquire about the status of an action or request using the confirmation ID. A web service inquiring on the status of a task can retrieve the confirmation details to display to the taxpayer. In addition, if the taxpayer has questions at a later date about a request, the confirmation ID may be used when contacting the tax authority about the request.

The common structure captured with each message (stored in the data area **C1-SelfServiceConfirmMessage**) is as follows:

- Confirmation ID. This may be generated prior to sending the web service to the product. For example, in many cases BPEL generates a confirmation ID and adds it to the message.
- Confirmation Header. This is populated with the message category / message number used as the acknowledgement message. For example: 'Your request has been submitted successfully, reference ID 1238098'. It is defined on the service task's task type.
- Confirmation Details. This is a collection of messages (message category / message number) that may be added while processing the transaction. For example: 'Tax Clearance Certificate approved', and 'Mailed to the taxpayer on 01-02-2012'

Task Confirmation Inquiry

A taxpayer may inquire on the status of a transaction or request using a confirmation number. If so, a web service is sent to the product. The product provides the inbound service **TSGetConfirmationInformation** to process this web service. It uses the script **C1-GetCnfmID** to process the request. This service script finds the service task associated with the confirmation ID, retrieves the confirmation details stored with the task, and returns the information to the calling system.

Confirmation Email

The product provides an algorithm to allow a tax authority to send an email to the taxpayer once a task that is performing maintenance actions has completed. For example, when a taxpayer submits a payment, the service task business object that processes the payment can be configured to send an email to the taxpayer once the payment is successfully created. The base algorithm includes the confirmation details from the service task in the email along with general email text that is configurable.

Fastpath: Refer to the algorithm section of *General Configuration Tasks* for more information about this algorithm.

Error Handling

All web services include standard error message elements. If any errors are found in processing a web service, a response is returned with the error message.

> **Note:** The system error messages may not be appropriate to display to the web self service user. The message can be translated to one that is more appropriate in the BPEL layer.

Configuration Errors

If the processing script used to process any of the messages finds a configuration problem or unrecognized information from the message, it returns a generic error to the self service application like "Your request can not be processed at this time, please try again later". In addition, a To Do entry and/or an email are sent to an appropriate responsible user.

> **Fastpath:** Refer to the algorithm section of *General Configuration Tasks* for more information about this algorithm.

Extending the Base Logic

All web services include a "custom information" area that allows an implementation to configure up to 10 additional fields using field name and value pairs that may be populated using a custom service script.

> **Fastpath:** Refer to the master configuration section of *General Configuration Tasks* for more information.

Standard Message Format

All web services which communicate with the self service application follow a common structure that includes the following information:

- Common data used to track the web user access details: User ID, Name, IP Address and email address. Note that in the base business objects this information is captured in the business object data area as well as in characteristics to allow searching by these values.
- Access keys. The web service supports capturing up to 10 key fields defined with field name and value pairs.
- Access type. In conjunction with access keys, the type identifies the tax account in the system.
- Error Details data area that contains an error message.
- A custom information area for implementation specific values.
- For the requests that perform transaction processing, there is also a standard data area that defines Confirmation Details.
- Additional optional elements included in multiple messages.
  - Action indicates what type of special processing is requested.
  - Linked request element contains the confirmation number of the related web service request.
  - Line of business.

Viewing Raw XML

The 'raw XML' of the web service is captured for audit purposes on any resulting service task. It's possible that a user may need to review the raw XML to troubleshoot problems. Because the data may contain personal and sensitive information from the taxpayer, such as a person identifier, the product provides additional security configuration for viewing the raw XML on service tasks.

Note: Refer to the business object descriptions for the base parent business objects **C1-StandardDeferredSSTask** and **C1-StandardRealTimeSSTask** for more information about securing this logic.

Settings in Master Configuration

Many implementation wide settings are needed when configuring the product to work with the web self service application. The product captures these settings in the Self Service Master Configuration record.

In the sections in this document that explain how to configure the system to use the base self service functionality, any settings in the self service master configuration record are noted.

Inquiry Audit

The product supports the ability for implementations to capture audit records of what data users are viewing. The integration with a self service application extends this capability so that audit records may be captured when the system receives requests for information from the self service application.

A special inquiry audit business object (**C1-WebAuditInquiry**) has been provided to capture audits of information requests from the self service product. The business object captures the web user id, the web user name, the email address, the IP address and, if configured, a snapshot of the actual request XML including any response.

Note: The user interface for this type of inquiry audit allows a user to view the request / response XML for those inquiry audit records that capture it. Because the data may contain personal and sensitive information from the taxpayer, such as a person identifier, the product provides additional security configuration for viewing the raw XML on inquiry audits. Refer to the business object description for the base business objects **C1-WebAuditInquiry** for more information about securing this logic.

Configuring Inquiry Audit

An implementation configures which web services should cause an inquiry audit record to be created. For example, perhaps web services for a refund status inquiry are audited, but web services for inquiring on the status of a task via the confirmation ID are not. The self service master configuration record allows an implementation to configure the web services that should be audited by indicating an appropriate audit service script to invoke when the web service is processed.

Fastpath: Refer to the master configuration section of *General Configuration Tasks* for more information.

Chapter 2

General Configuration Tasks

This section describes the general tasks required to configure the system to support the integration. It also describes the service task type portal.

Refer to the Supported Functionality section for addition details about configuration required for specific business functionality.

Messages

Review the base messages provided for communication of positive and negative responses to the self service user. These are configured on self service task types.

- Confirmation Header message. The product provides a base message that may be used, with message category 11126 and message number 11723. The base message text may be overwritten. Or a new message in a message category reserved for implementations may be created.
- Error Header message. The product provides a base message that may be used, with message category 11126 and message number 11010. The base message text may be overwritten. Or a new message in a message category reserved for implementations may be created.

Note: In the BPEL layer the product's Message Category / Message Number combination is translated into message codes understood by the self service product.

Managed Content

Create a managed content record to define the HTML to use for the various general emails.

- Create one for emails routed to a responsible user when problems are found with the master configuration table.
- If confirmation emails should be sent to taxpayers when tasks with multiple steps are complete, create a managed content entry for that email text. If different service tasks should have different general text in the confirmation email, create a separate record for each variation.

The Managed Content Type should be HTML. Navigate to the Schema tab and enter the following tags and type the appropriate text for the email within the "<span>" tags.

```
<html>
```

```
<body>
<span> </span>
</body>
</html>
```

XAI Sender

Verify that an XAI sender is configured for routing email real time.

- Invocation Type should be set to **Real-time**
- XAI Class should be set to **RTEMAILSNDR**
- On the Context tab, the context type **SMTP Host name** should be defined with a valid context value.

> **Note:** The XAI Options includes an option for a default Sender for real-time emails that can be configured with this XAI sender.

Field

Define a field that includes the text for the email subject for the notification of master configuration errors email.

Define one or more fields that include the text for the email subject for the confirmation emails to send for the various service tasks.

Algorithm

If any service task should send a confirmation email after all steps are complete, for example once a payment is processed, configure one or more appropriate algorithms for the **C1-SSCONFREM** algorithm type.

- Set the Email Subject field created above for the confirmation email.
- Set the Predefined Text to the Managed Content created above for the confirmation email.
- Enter the From Address that should be used in the generated email
- Enter the XAI Sender for Email defined above.

> **Note:** This algorithm must be plugged into appropriate business objects as an enter algorithm on a state where actions done by the task are complete.

Access Type

Base product is provided with the implementation of access type *Tax Role* with keys **PER_ID** and **TAX_ROLE_ID**. If required, additional *access types* should be added to the lookup **USER_ACCESS_TYPE_FLG**.

Line of Business

Base product is provided with the implementation two Lines of Business: **Individual** and **Business**. If required, additional *line of business* entries should be added to the lookup **LINE_OF_BUSINESS_FLG**.

Self Service Master Configuration

Create a Self Service Master Configuration record that holds system wide configuration needed for the self service integration.

The Configuration Support section defines information needed to alert a user that problems were found with configuration. The alert may be sent using a To Do entry or an email or both.

- Define the user that is responsible for technical issues related to self service who should receive an email if there is an issue.

- To Do Type should be set to a standard error To Do type to use when issues are found with the master configuration table. The base product To Do Type **Self Service Master Configuration Issues (C1-SSMCI)** is provided for this purpose.
- To Do Role should be configured to an appropriate default To Do role for the above To Do type.

Populate the following fields if an email should be generated when configuration problems exist.

- Enter the From Email Address that should be used in the email.
- Enter the From Name that should be used in the email.
- Enter the XAI Sender created above or leave it blank to use the default from XAI options.
- Set the Email Content to the Managed Content record created above.
- Set the Email Subject field created above.
- Provide the Configuration Support Email Script that produces the email. The product has supplied **C1-SSIssueEm** which may be used here.

The Request Processing Control information allows an implementation to define special logic that should occur when a web service is processed. The special logic may be causing an inquiry audit record to be created or it could be to indicate a custom extension to a base product service by retrieving additional details to return in a web service call.

- To Do Type for Audit Script Errors should be set to an appropriate To Do type. The base product To Do Type **Self Service Task Audit Issues (C1-SSAI)** is provided for this purpose.
- To Do Role for Audit Script Errors should be configured to an appropriate default To Do role for the above To Do type.
- To Do Type for Custom Extension Errors should be set to an appropriate To Do type. The base product To Do Type **Self Service Task Custom Extension Issues (C1-SSCSI)** is provided for this purpose.
- To Do Role for Custom Extension Errors should be configured to an appropriate default To Do role for the above To Do type.

For any web service that requires inquiry audit or custom extension, determine the schema associated with the XAI inbound service that processes the web service.

- Schema Type and Schema Name should be set to the appropriate object that the XAI inbound service references.
- For enabling inquiry audit, check the Inquiry Audit Enabled button. If the raw XML of the request and response should be captured, check the Store Response Data checkbox. Indicate the Audit Script. The base product provides the script **C1-AddWebAI** that may be used.
- For providing additional data in a web service response, provide a Custom Extension Script. The schema for the custom extension script should match the schema that it is extending. It populates data in the data area **C1-SSCustomExtensionFields**.

Setting Up Taxpayer Identification

Base product functionality uses a person's birth date as one of the identification details. If your implementation captures the birth date on the person record and would like that to be part of the taxpayer identification logic, the setup described in this topic is required.

Note: The product does not provide any functionality for populating the birth date characteristic on the person.

Characteristic Type

Create a business object for individual person that supports display and maintenance of the birth date characteristic.

Business Object for Individual Person

Create a business object for an individual person that supports display and maintenance of the birth date characteristic:

1. Add a child business object for **C1-PersonIndividual**.
2. Copy the schema.
3. Add the element representing a flattened characteristic of the type created above, placing it in the desired location for the rendered user interface. In addition, include the appropriate **startSection** and **endSection** definitions.

The following example illustrates the schema for a child business object created for the base business object **C1-PersonIndividual** and characteristic type **BIRTHDAT** so that the birth day appears in a new ***Additional Information*** section after the ***Contact Information*** .

```
<schema xmlns:uiHint="http://oracle.com/ouafUIHints">
<includeDA name="C1-PersonCommon"/>
<includeDA name="C1-PersonExtendedNameDetails"/>
<includeDA name="C1-PersonId"/>
<includeDA name="C1-PersonPhone"/>
<includeDA name="C1-PersonAddress"/>
<includeDA name="C1-PersonContactInfo"/>
<uiHint:startSection mdField="ADDITIONAL_INFO" sectionColumn="right"/>
<birthDate mdField="C1_SS_BIRTHDATE" dataType="date" mapField="ADHOC_CHAR_VAL">
<row mapChild="CI_PER_CHAR">
<CHAR_TYPE_CD is="BIRTHDAT"/>
<EFFDT is="%effectiveDate"/>
</row>
</birthDate>
<effectiveDate dataType="date" default="1950-01-01" mapField="EFFDT" rowRef="birthDate"
 suppress="true"/>
<uiHint:endSection/>
<includeDA name="C1-PersonPerson"/>
</schema>
```

Person Type

Specify the business object created above on person type(s) that represent individuals in your implementation.

Setting Up Service Task Types

Service Task Types contain the rules that control how service tasks are processed.

To set up a service task type, select **Admin Menu** > **Self Service Task Type** .

The topics in this section describe the base-package zones that appear on the Service Task Type portal.

Self Service Task List

The Service Task Type List Zone lists every self service task type. The following functions are available:

- Click the broadcast button to open other zones that contain more information about the adjacent service task type.
- The standard actions of **Edit, Delete** and **Duplicate** are available for each service task type.
- Click the **Add** link in the zone's title bar to add a new service task type.

Self Service Task Type

The Self Service Task Type zone contains display-only information about a service task type. This zone appears when a service task type has been broadcast from the Self Service Task Type List zone or if this portal is opened via a drill down from another page.

Please see the zone's help text for information about this zone's fields.

Chapter 3

Supported Functionality

This section describes the out of the box functionality provided for the integration along with the required configuration tasks.

Enrollment

After registering on a self-service portal and establishing a secure login, the user should explicitly request access to his/her taxes. The request is evaluated, and the user/tax account association(s) are recorded in the system and become available for account access verification.

A single web portal login provides the self service user with access to all tax accounts. In other words an individual who also happens to be a business owner may log in once and view and manage both individual and corporate taxes.

A user enrolled into a tax account is able to view account financial information, file tax returns and make payments related to this account, update contact and correspondence information, and request services.

Note: See the *Tax Account Access Information* section for a detailed explanation about tax account access.

User Access Store

The User Access store is a table in which the link between users and tax accounts is stored. It is maintained and accessed exclusively from the integration layer via SOA composites. The integration provides the information to the self service product and other solution components.

- The following information is stored:
- Enrollment ID
- Web User ID
- Line of Business

- Access Type
- Access keys 1 - 10 (Key Name and Key Value)
- Revenue Management System Name
- Status

The records marked with same enrollment ID/web user ID/line of business comprise an enrollment "event".

The possible statuses of the user access records are represented in the product by the lookup **ENROLLMENT_STATUS_FLG.**

Initial Enrollment and Enrollment Event

The self service user is asked to provide identification details sufficient to determine tax accounts that this user owns. A separate enrollment request is made for each line of business.

The initial enrollment request creates an enrollment "event" that is "known" to two solution components: the integration layer and the revenue management system(s). Service task represents an enrollment event in the base product. User account access data captured in the user access store includes enrollment ID, web user ID, and line of business, thus associating it with a specific enrollment event.

The solution supports configurable identification details and algorithm-based enrollment request processing. In the base product, initial enrollment is implemented using an *Enrollment Service Request*.

The request is handled by a special web service TSEnrollmentServiceRequest. The results of a successful enrollment are:

- Creation of one or more records in the user access store. Each record is associated with enrollment ID. One record is created for each accessible tax account. The status of the record indicates that the access to a tax account is either approved or requires an additional review.

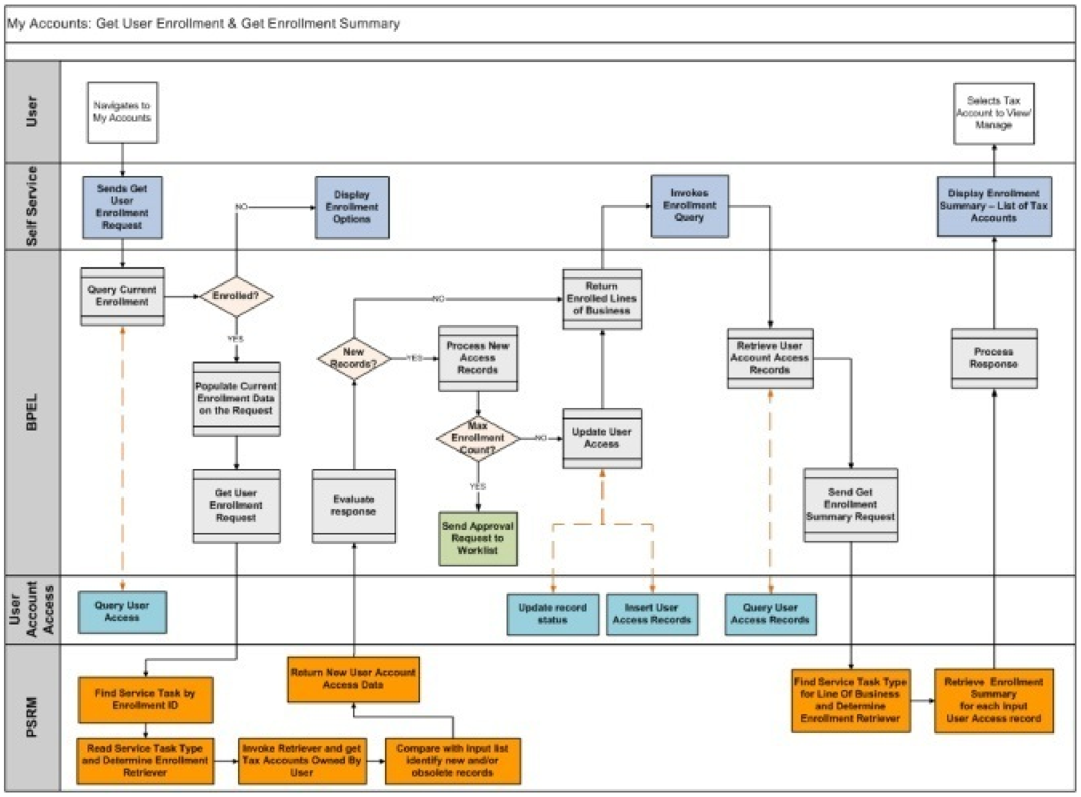
  Note: The status of the record is set according to service task type configuration.

- Creation of a service task associated with both enrollment ID and confirmation ID. The initial list of user access records is captured on the service task.

Note: See the *Enrollment Request* section for detailed information about enrollment processes and related configurations.

Enrollment Refresh and Summary

The following diagram illustrates the process flow on the **My Accounts** (Enrollment Summary) portal page.


My Accounts: Get User Enrollment & Get Enrollment Summary
User
Self Service
BPEL
User Account Access
PSRM
Navigates to My Accounts
Selects Tax Account to View/ Manage
Sends Get User Enrollment Request
NO
Display Enrollment Options
Invokes Enrollment Query
Display Enrollment Summary – List of Tax Accounts
Query Current Enrollment
Enrolled?
YES
NO
Return Enrolled Lines of Business
New Records?
YES
Process New Access Records
Retrieve User Account Access Records
Process Response
Populate Current Enrollment Data on the Request
Max Enrollment Count?
NO
Update User Access
YES
Get User Enrollment Request
Evaluate response
Send Approval Request to Worklist
Send Get Enrollment Summary Request
Query User Access
Update record status
Insert User Access Records
Query User Access Records
Return New User Account Access Data
Find Service Task by Enrollment ID
Find Service Task Type for Line Of Business and Determine Enrollment Retriever
Retrieve Enrollment Summary for each Input User Access record
Read Service Task Type and Determine Enrollment Retriever
Invoke Retriever and get Tax Accounts Owned By User
Compare with input list identify new and/or obsolete records

Implicit Enrollment Refresh

User access information is implicitly refreshed upon login. For example, a business owner that opened a new business and registered it with the tax authority would be automatically given an access to this new business account.

The request is handled by a special web service, **TSGetUserEnrollment**.

The integration has the following steps:

- The request is initiated from the self service product. The XML message contains web user ID.
- The SOA Composite is querying the user access store, retrieves all enrollment records and populates them on the request; each record includes tax account identifiers line of business and enrollment ID.
- The request is forwarded to the product. The processing service script finds enrollment request service task associated with each input enrollment ID. It reads service task type configuration under *Enrollment Instructions*, and invokes the enrollment retrievers associated with the input access type. The input list of tax accounts is compared with the retriever's response, and any new entries are added to the output. Entries that are no longer present on the list are added to the output with *Inactive* status.

  Note: See *New and Updated User Access* for more information. Also refer to the business service **C1-GetEnrollment** for additional details.

- The response is processed by the integration layer and new user's tax accounts are added to the user access store.

The results of a successful enrollment refresh may be a creation of one or more new records in user access store. Each record is stumped with the original enrollment ID/line of business combination.

Enrollment Summary

Upon login the self service product displays the user enrollment summary that contains a list of all tax accounts that the user is eligible to view and manage. This is an inquiry request that defines its own web service (**TSGetEnrollmentSummary**). The integration has the following steps:

1. The request is initiated from the self service product. The XML message contains the user ID.
2. The SOA Composite queries the user access store and retrieves all enrollment records. Each record includes tax account identifiers (access type and access keys), line of business, and enrollment ID.
3. The list of enrollment records is delivered to the system. The processing service script finds the enrollment service task associated with the input enrollment ID, reads the service task type, and calls the account summary retriever. The retriever also marks one enrollment record as a default. When the web service response reaches portal application, this record is automatically pulled into a self service session context.

Summary Contents

The self service product displays tax account highlights on the **My Accounts** portal page. The single tax account enrollment summary includes two components:

- **Title** - a short description of the tax account.
- **Details** - essentials about tax account status.

Both parts of the summary are composed using message text with the substitution parameters.

The base product is expected to supply the actual data items, including dates, numbers, and types. This information is used by the self service product as substitution parameters for the Summary messages.

The enrollment summary retriever is expected to deliver the following information:

- Summary Title parameters list.
- Summary Details parameters list.
- Taxpayer Name (required, used internally by the self service product).
- Account Name (optional, used internally by the self service product).

The enrollment summary retrievers provided with the base product return the following

- **Individual** enrollment summary - returns a summary title parameter *tax type* and summary details parameters: tax role's *start* and *end date* and an *outstanding balance* forecasted for the current date.
- **Business** enrollment summary - returns a summary title parameter *tax type* and summary details parameters: tax role's *start* and *end date* and an *outstanding balance* forecasted for the current date and a *formatted address*.

Note: Refer to the business service **C1-GetBusinessTaxRoleSummary** as an example of a tax account enrollment summary retriever.

New and Updated User Access

Enrollment retriever logic compares the result list of tax accounts with the initial enrollment data that is captured on the service task and performs the follows:

- Adds the new entries to the output new keys list.
- If an entry exists on the initial enrollment list but no longer returned by the retriever, the logic interprets it as if the taxpayer is no longer owns the tax account. Such record is added to the new keys list with an **Inactive** status.

Note: Refer to the business service **C1-EnrolIndividualAccntTaxRole** as an example of an access type-specific enrollment retriever.

Failed Enrollment Attempts

The self service solution provides a mechanism to monitor enrollment attempts and block suspicious and fraudulent web user activities.

A service task "in error" is created every time an enrollment process ends in error. This allows tracking and auditing of the failed attempts. The results of unsuccessful enrollment attempt are:

- A service task in error is created. It is associated with the enrollment ID.
- The preceding failed enrollment attempts determined as service task(s) in error with the same enrollment ID. If found, they are linked to the current service task as related objects.
- A single record in error status is created in the user access store (this is handled by a SOA Composite in the integration layer).

Enrollment Review

The self service product allows manual review of the problematic enrollment requests. Under certain conditions the SOA Composite may place one or more records in the user access store on hold and create a **Worklist** task.

See the *Oracle Public Sector Revenue Management Self Service Implementation Guide* for more information about how problem enrollment requests are handled.

Configuration Tasks for Enrollment

The following sections list the configuration tasks required to implement the base user enrollment functionality.

Master Configuration

The self service master configuration record includes settings required for enrollment processing.

- **Revenue Management System**. Define the alphanumeric code that will be referenced by records in the user access store.

  Note: See the *User Access Store* section for more details.

Service Task Type

Configure Enrollment Request service task type for every line of business supported by your implementation. This service task type contains definitions used by *Initial Enrollment Request*, *Implicit Enrollment Refresh* and *Enrollment Summary* functionality.

See *Configuration Tasks for Enrollment Request* for detailed information on enrollment configurations.

Integration Mapping

Tax types and **taxpayer types** used by the product should have corresponding values defined in the self service product.

Your implementation may decide to use exactly the same values in both products or configure the mapping using domain value mapping in the SOA Composer application

Customizing Enrollment

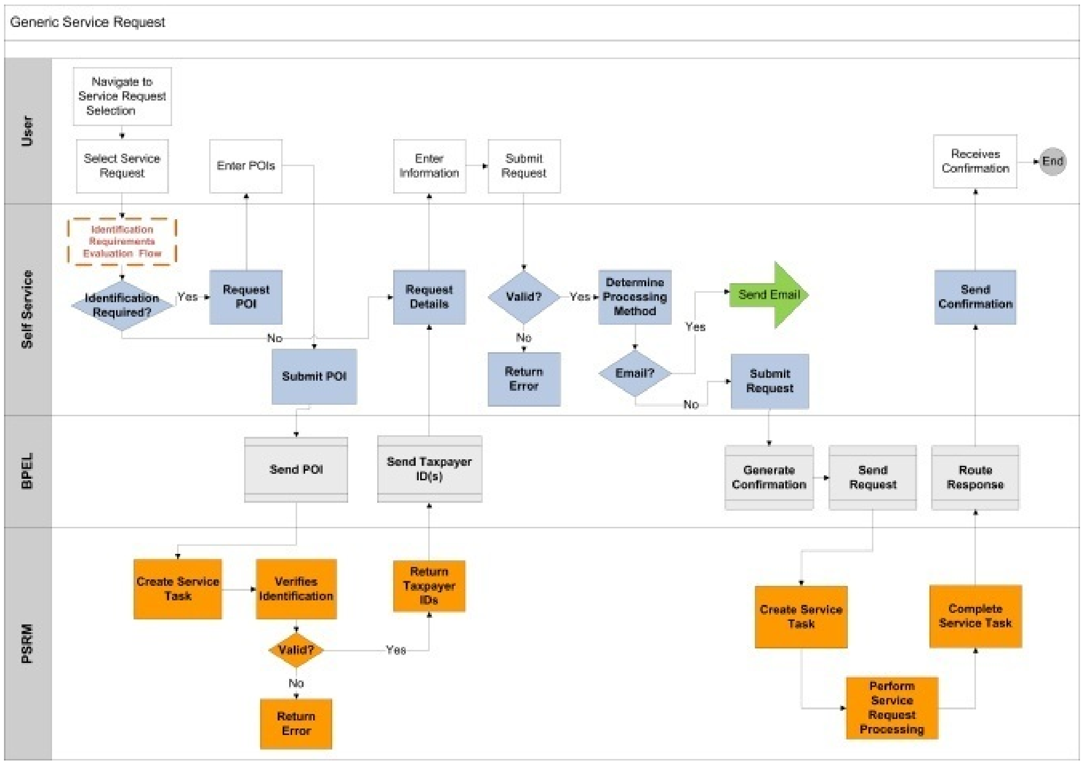
If your business requirements are not satisfied by the functionality provided by the base product consider the following customization options:

- **The information displayed on My Accounts**. To customize the contents of a tax account summary you should create custom Access Type Summary Retriever and adjust the enrollment request service task type configurations and also the corresponding configurations in self service product (message).
- **The way tax accounts owned by the user are determined**. Create custom User Access Type Summary retriever(s) and adjust the corresponding enrollment request service task type configurations.
- **The information required in order to verify self service user identity and credentials** If different from what is supported by a base product, you should create new enrollment service task business object to capture the information and implement an alternative User Enrollment Retriever(s). Reconfigure the corresponding enrollment service task types to reference the new business object and set up new request field mapping.
- **Additional steps should be included in the initial enrollment handling**. Utilize standard business object plug in spots available the enrollment service task.
- **New access type**. Implementation of a new access type should be closely coordinated with the self service product.
  - Define access keys to be used for the new access type. **Note**: key 1 is always reserved for the Taxpayer ID.
  - Determine the lines of business that are applicable for this access type.
  - Implement User Enrollment Retrievers and Access Type Summary Retrievers for all applicable lines of business.
  - Configure enrollment service task types.
  - If the implementation should support an option to pay a balance for the tax account defined by this new access type, implement a new payment destination, including distribution rule, if needed.
  - Create corresponding configurations for Access Type in the self service product and add mapping in the integration layer

Note: See *Configuration Tasks for Enrollment Request* for detailed information on enrollment configurations.

Taxpayer Service Request

The self service product supports service requests by the taxpayer. Some requests may be fulfilled by the self service product, such as information about when to file or technical support requests. Other requests may require information from this product. The following diagram illustrates an expected process flow.



The following points highlight the process:

- The service request may or may not require taxpayer identification up front. If it does, the initial processing works as follows:
  - **For a casual user**: The self service application captures information about the taxpayer and sends a web service request to this product to validate the taxpayer (proof of identity - POI).
  - **For an enrolled user**: The identification is not performed.

  Note: For additional information, see *User Identity Verification*.

- The product receives a web service request to identify the taxpayer. The product provides the inbound web service **TSTaxpayerIdentification** to process the message. It uses the script **C1-IdentTaxp** to process the request. This should cause an appropriate service task to be created based on the type and populate the relevant data for the service task. The service task should include algorithms to identify the taxpayer and immediately respond to the web service call.

  Note:
  For additional information see *Configuration Tasks for Taxpayer Service Request*.

- For any service request that requires action by the product, another web service request is received with information about the request, including the task type to create. The product provides the inbound web service **TSTaxpayerServiceRequest** which uses the script **C1-TaxSvcReq** to process the request. This should cause an appropriate service task to be created based on the type and populate the relevant data for the service task. Depending on the requirements for a particular service request, the service task may be designed to do the following:
  - Give real-time feedback/results (if the request simply involves information retrieval)
  - Perform the task as a separate step without the taxpayer waiting for a reply. In this case the taxpayer should receive a confirmation ID related to the request that is provided in the web service so that at a later date the progress of the service request can be queried by the taxpayer.

  Note: See *Implementing A New Service Request* for more information.

The base product provides functionality out of the box for various categories of service requests, including a taxpayer's request for a tax clearance certificate. The topics below describe more information about the provided functionality, followed by configuration tasks.

Configuration Tasks for Taxpayer Identification Service Request

The following sections list the configuration tasks required to implement the base taxpayer identification functionality.

Algorithms

Create the following algorithms:

- Create an algorithm for algorithm type **C1-IDTXPSR**. Populate the Birth Date Required parameter as per business rules. Define the characteristic type for the birth date that was created above (also see *Setting Up Taxpayer Identification* in the topic, *General Configuration Tasks*.

Business Object

Update the business object **C1-TaxpayerIdentSvcReqSSTask** to include this algorithm. Navigate to the Lifecycle tab and on the **In Progress** state, add an entry to the Algorithm collection: System Event is **Enter**, Algorithm is the one created above.

Service Task Type

Create a service task type for taxpayer identification to use for service requests. Use the business object **Taxpayer Service Request Self Service Task Type**.

- The related transaction BO should be set to **C1-TaxpayerIdentSvcReqSSTask**.
- Service task class should be set to **Service Request**.
- To Do Type should be set to a standard error To Do type to use when transitioning to error. The base product To Do Type **Self Service Task Issues (C1-SSTTD)** is provided for this purpose.
- To Do Role should be configured to an appropriate default To Do role for the above To Do type. This is optional. If no value is provided the default role for the To Do type is used.
- Configure the Confirmation Header Message Category and Message Number and the Error Header Message Category and Message Number to use for communication to the self service user.

  Fastpath: Refer to the Messages section of the *General Configuration Tasks* for more information.

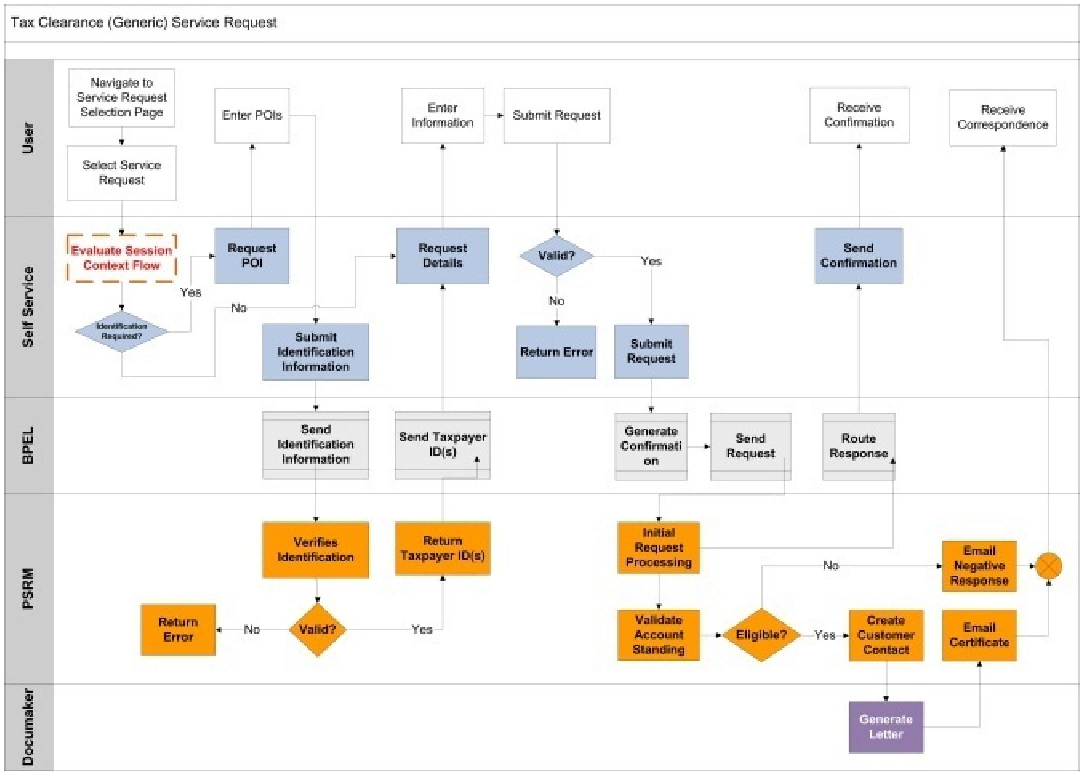
- The service request fields mapping is used when creating the target service task to correctly populate the requestField list from the list of fields passed in on the web service request XML. For the base transaction business object, the following fields are expected:

Sequence	Transaction BO XPath
1	requestData/legalName
2	requestData/birthDate
3	requestData/personType
4	requestData/idType
5	requestData/personIdNumber

Fastpath: Service Task Type Mapping. The service task type defined here must be configured in the service task domain value mapping in the SOA Composer application. Refer to the self service product documentation for details.

Tax Clearance Certificate

On occasion a taxpayer may be required to produce a Tax Clearance Certificate to a third party. This must be requested from the tax authority as a written confirmation that a taxpayer is considered in good standing as of the date of issue of the Certificate. The following diagram illustrates the process flow supported for issuing a tax clearance certificate.



The following points describe in more detail the functionality provided

- This type of service request requires taxpayer identification first.
- The product provides a base service task business object called **C1-TaxClearCertSvcReqSSTask** that processes the request. It calls an algorithm to determine if the account is in good standing. If so, it creates a customer contact to issue the certificate that may be emailed. If not, it creates a customer contact to inform the taxpayer that a certificate is not possible.
- The product provides a letter extract for a sample tax clearance certificate. In addition a letter template for Documaker is included in this functionality.

Configuration Tasks for Tax Clearance Certificate

The following sections list the configuration tasks needed to implement the tax clearance certificate functionality.

Lookup

Using the lookup **C1_TAX_CLEARANCE_PURPOSE_FLG** define the values that are valid for your implementation.

Characteristic Types

Create the following characteristic types:

- **Tax Clearance Purpose**. This char type is used to capture the tax clearance purpose on the customer contact for audit purposes. Define Customer Contact as a characteristic entity. The tax clearance purpose is defined in a lookup field (**C1_TAX_CLEARANCE_PURPOSE_FLG**). The characteristic type should be defined as a predefined characteristic with the valid values from the lookup repeated.

Letter Template

Create a letter template to represent the tax clearance certificate. Reference the extract algorithm **C1-LTR-TCSDC**. Batch control is required and **LTRPRT** can be entered here.

Customer Contact Class / Type

Define or identify an appropriate Customer Contact Class for the tax clearance certificate customer contact types.

Create a Customer Contact Type for issuing the tax clearance certificate.

- Customer Contact Business Object is not required.
- Contact Shorthand is not applicable.
- Set the Contact Action to **Send Letter** and indicate the letter template created above.
- For the template characteristics, define the Characteristic type created above for Tax Clearance Purpose. In addition, define the (base) characteristic type Expiration Date.

Create a Customer Contact Type for declining the Tax Clearance Certificate.

- Customer Contact Business Object is not required.
- Contact Shorthand is not applicable.
- Leave Contact Action blank.
- For the template characteristics, define the Characteristic type created above for Tax Clearance Purpose.

Managed Content

Create managed content records to define the HTML to use for the email related to the tax clearance certificate. Define one for the email that is sent when issuing the certificate. Define another for the email that is sent when the request is denied.

For both records, the Managed Content Type should be HTML. Navigate to the Schema tab and enter the following tags and type the appropriate text for the email within the "<span>" tags.

```
<html>
<body>
<span> </span>
</body>
</html>
```

Field

Define a field that includes the text for the email subject.

Algorithms

Create the following algorithms:

- Evaluate Outstanding Debt. Create an algorithm for the algorithm type **C1-EVALDEBT**. Populate the Max Outstanding Debt Allowed parameter.

- Evaluate Gap in Filing. Create an algorithm for the algorithm type **C1-EVALGAP**. Populate the Overdue Process Template and the No-gap Filing Period parameters.
- Email Tax Clearance Certificate Letter. Create an algorithm for the algorithm type **C1-EMTXCL**. Populate the parameters:
  - Set the Customer Contact Class created above.
  - Set the Customer Contact Types for Issue Tax Clearance Certificate and for Decline Tax Clearance Certificate created above.
  - Set the Characteristic Type for Tax Clearance Certificate Purpose
  - Set the Predefined Email Text for the Issuing Certificate Email and the Decline Request Email configured as Managed Content above.
  - Set the Email Subject field created above.
  - Enter the Characteristic Type for Certificate Expiration Date: C1-EXPDT
  - Enter the number of days that is the Certificate Valid Period
  - Enter the XAI Sender for Email created as part of the general configuration options.
  - Enter the From Email Address that should be used in the generated email
  - Enter the From Name that should be used in the generated email

Business Object

Update the business object **C1-TaxClearCertSvcReqSSTask**. Navigate to the Lifecycle tab.

- On the **In Progress** state, add an entry to the Algorithm collection: System Event is **Enter**, Algorithm is set to the one created above for Evaluate Outstanding Debt.
- On the **In Progress** state, add an entry to the Algorithm collection: System Event is **Enter**, Algorithm is set to the one created above for Evaluate Gap in Filing.
- On the **Complete** state, add an entry to the Algorithm collection: System Event is **Enter**, Algorithm is set to the one created above to Email the Tax Clearing Certificate Letter.

Service Task Type

Create a service task type for the tax clearance certificate service task with the business object **Taxpayer Service Request Self Service Task Type**.

- The related transaction BO should be set to **C1-TaxClearCertSvcReqSSTask**.
- Service task class should be set to **Service Request**.
- To Do Type should be set to a standard error To Do type to use when transitioning to error. The base product To Do Type **Self Service Task Issues (C1-SSTTD)** is provided for this purpose.
- To Do Role should be configured to an appropriate default To Do role for the above To Do type. This is optional. If no value is provided the default role for the To Do type is used.
- Configure the Confirmation Header Message Category and Message Number and the Error Header Message Category and Message Number to use for communication to the self service user.

  Fastpath: Refer to the Messages section of the *General Configuration Tasks* for more information.

- The service request fields mapping is used when creating the target service task to correctly populate the requestField list from the list of fields passed in on the web service request XML. For the base transaction business object, the following fields are expected:

Sequence	Transaction BO XPath

1	requestData/emailAddress
2	requestData/phoneNumber
3	requestData/taxClearancePurpose
4	requestData/isPaperCopyReq

Fastpath: Service Task Type Mapping. The service task type defined here must be configured in the service task domain value mapping in the SOA Composer application. Refer to the self service product documentation for details.

Scheduled Reminder

A self-service user may schedule an automatic email reminder. The reminder can be scheduled for an infinite time or expire on a certain date. The monthly reminder functionality is provided by the product. This is a standard taxpayer service request.

Note: See *Taxpayer Service Request* for details on the request flow.

The following points describe the functionality provided:

- This type of service request does not require taxpayer identification.
- The product provides a base service task business object called C1-SchedMonthlyReminderSSTask that processes the request. It creates a reminder service task in active state. Periodic monitor batch process selects the active reminder tasks and triggers an algorithm to determine if the expiration date is reached. If not yet, the subsequent algorithm checks if the scheduled day of the month is reached and if so, sends an email reminder. If the expiration date id reached or in the past, the reminder task becomes inactive.

Configuration Tasks for Scheduled Reminder

The following sections list the configuration tasks needed to implement the scheduled reminder request functionality.

Managed Content

Create managed content records to define the HTML to use for the email reminder.

The Managed Content Type should be HTML. Navigate to the Schema tab and enter the following tags and type the appropriate text for the email within the "<span>" tags.

```
<html>
<body>
<span> </span>
</body>
</html>
```

Field

The product is provided with field C1_SCHED_REMINDER_SUBJ Tax Payment Reminder. Use it for the reminder email subject or define a new field that includes the text for the email subject.

Service Task Type

Create a service task type scheduled monthly reminder request. Using the business object Scheduled Reminder Request Task Type:

- The related transaction BO should be set to C1-SchedMonthlyReminderSSTask.
- Service task class should be set to Service Request for To Do.
- To Do Type should be set to a standard error To Do type to use when transitioning to error. The base product To Do Type Self Service Task Issues (C1-SSTTD) is provided for this purpose.
- To Do Role should be configured to an appropriate default To Do role for the above To Do type. This is optional. If no value is provided the default role for the To Do type is used.
- Configure the Confirmation Header Message Category and Message Number and the Error Header Message Category and Message Number to use for communication to the self service user.

  Note: Fastpath: Refer to the Messages section of the *General Configuration Tasks* for more information.

- The service request fields mapping is used when creating the target service task to correctly populate the requestField list from the list of fields passed in on the web service request XML. For the base transaction business object, the following fields are expected:

Sequence	Transaction BO XPath
1	requestData/name
2	requestData/emailAddress
3	requestData/phoneNumber
4	requestData/dayOfMonth
5	requestData/expirationDate

Configure Email Notification Instructions:

- Set the Email content configured to Managed Content created above.
- Set the Email Subject to Field created above.
- Enter the From Email Address that should be used in the generated email
- Enter the From Name that should be used in the generated email
- Enter the XAI Sender for Email created as part of the general configuration options

Note: Service Task Type Mapping. The service task type defined here must be configured in the service task domain value mapping in the SOA Composer application. Refer to the self service product documentation for details.

Generic To Do Service Request

Some services requested by the taxpayer may require manual review and idiosyncratic processing. The integration supports the scenario where the incoming service request results in creation of a service task which simply stores request fields (name-value pair collection) and creates a To Do entry that is referencing the service task.

Note: See *Taxpayer Service Request* for details on the request flow.

This approach allows utilizing single service task business object to capture different types of requests.

Configuration Tasks for Generic To Do Request

The following sections list the configuration tasks needed to implement the generic service request functionality.

Service Task Types

Create a service task type for each generic create-To-Do service request. Use the business object Generic Create To Do Request Task Type.

- The related transaction BO should be set to C1-GenericToDoSSTask.
- Service task class should be set to Service Request for To Do.
- To Do Type should be set to a standard error To Do type to use when transitioning to error. The base product To Do Type Self Service Task Issues (C1-SSTTD) is provided for this purpose.
- To Do Role should be configured to an appropriate default To Do role for the above To Do type. This is optional. If no value is provided the default role for the To Do type is used.
- Configure the Confirmation Header Message Category and Message Number and the Error Header Message Category and Message Number to use for communication to the self service user.

    Fastpath: Refer to the Messages section of the *General Configuration Tasks* for more information.

- Request processor should be set to C1-PopSvRqDt.
- The service request fields mapping is used when creating the target service task to correctly populate the requestField list from the list of fields passed in on the web service request XML. For the base transaction business object, the following fields are expected:

Sequence	Transaction BO XPath
1	requestData/requestComments

To configure request processing instructions:

- Create To Do Type should be set to a To Do type to use when processing the request. The base product provides Generic Service Request To Do (C1-GSRTD) for this purpose.

    Note: To distinguish between various generic requests you may prefer to create a designated To Do Type for each one.

- Create To Do Role should be configured to an appropriate default To Do role for the above To Do type. This is optional. If no value is provided the default role for the To Do type is used.

Note: Service Task Type Mapping. The service task type defined here must be configured in the service task domain value mapping in the SOA Composer application. Refer to the self service product documentation for details.

Refund Status

In this release, the integration supports the ability for a taxpayer to inquire on the status of a tax refund. This is a special kind of taxpayer service request that defines its own web service. The integration has the following steps:

- The self service application captures information about the taxpayer, a way of identifying the filing period for the refund in question and a "shared secret" that ideally only the taxpayer and the tax authority would know (such as the expected refund amount).
- This information is sent to the system, which creates a Refund Status Inquiry service task.
- The task's lifecycle is designed to process the information immediately with no deferred monitor process so that the information can be returned to the self service application real-time.
- The base business object for the Refund Status Inquiry service task includes an algorithm to verify the taxpayer identification, confirm that a form has been received for the filing period, determine the status of the refund and compose an appropriate message to send back to the taxpayer immediately.

Note that the implementation of the refund status inquiry creates a service task for the following reasons:

- The creation of a service task with its business object architecture provides many plug-in spots to allow implementations to easily customize the logic for identifying the taxpayer and determining the refund status.
- The status of a refund is based on conditions in the system that can change over time. Instantiating a service task for the refund status inquiry provides a way of capturing the point in time that the request was received and being able to reconcile this with the information sent to the taxpayer.
- Creating a service task enables the tax authority to provide the taxpayer with a confirmation ID for possible follow up in the future. For example, imagine a taxpayer is told "Your refund has been processed. You should receive it shortly." and several weeks later the taxpayer has still not received the refund. If the taxpayer contacts the tax authority and provides the confirmation ID, the user can view the service task, verify the details and more easily investigate the situation.

The topics below describe more information about the provided functionality, followed by configuration tasks.

Refund Inquiry Service Task

The web service requesting the refund status is processed by the XAI Inbound Service **TSGetRefundStatus**, which invokes the service script **C1-RSISvcReq**. The appropriate service task type to use must be passed in with the other information received from the self service system.

Identifying the Shared Secret Amount

The logic provided in the base product assumes that the shared secret is an amount on the taxpayer's form. Form definition in the system is configurable and each form may be defined with different form lines that capture the amount used for the shared secret. For each form type, the system needs to know which form line is the one used for the shared secret. This information is defined in the self service master configuration.

Fastpath: Refer to *Configuration Tasks for Refund Status* for information about configuring the system to support these payment destinations.

Determining the Status of the Refund

The lifecycle of creating a tax refund in the system includes the following steps.

- The taxpayer files the form, which is received by the system. The form gets processed and appropriate adjustments are created for the obligation for the filing period to record the affect on the balance.
- Once the taxpayer's obligation has a credit balance, an overpayment process is used to process the refund. How the overpayment process is created depends on the business practice. An account debt monitor may be used to detect obligations with a credit and can create the overpayment process. The form can include logic to immediately create the overpayment process based on the existence of a credit balance based on a form rule.

  Note: The base product does not currently supply a form rule to create the overpayment process at posting time. A custom form rule may be developed.

- The overpayment process may require approval. Once approvals are processed, the overpayment process may have logic to use the credit to apply to other debt. Or it may carry the credit forward to a future filing period (typically upon the taxpayer's request). When the overpayment process completes, it has a record of carry forward, offset and refund amounts. If there is any refund, the overpayment creates an accounts payable (A/P) adjustment if the refund should be provided using a paper check. It creates a credit payment with appropriate bank information to process the refund as a direct deposit.

An algorithm plugged into the refund status inquiry service task business object as an enter plug-in on the **In Progress** state is responsible for determining the status of the refund. Refer to the description of the algorithm for more information about the base logic provided.

Configuration Tasks for Refund Status

The following sections list the configuration tasks needed to implement the refund status functionality.

Service Task Type

Create a service task type for the refund status inquiry service task with the business object **Standard Self Service Task Type**.

- The related transaction BO should be set to **C1-RefundStatusInquirySSTask**.
- To Do Type should be set to a standard error To Do type to use when transitioning to error. The base product To Do Type **Self Service Task Issues (C1-SSTTD)** is provided for this purpose.
- To Do Role should be configured to an appropriate default To Do role for the above To Do type. This is optional. If no value is provided the default role for the To Do type is used.
- Configure the Confirmation Header Message Category and Message Number and the Error Header Message Category and Message Number to use for communication to the self service user.

  Fastpath: Refer to the Messages section of the *General Configuration Tasks* for more information.

Fastpath: Service Task Type Mapping. The service task type defined here must be configured in the service task domain value mapping in the SOA Composer application. Refer to the self service product documentation for details.

Master Configuration

The self service master configuration record includes several settings required for refund status functionality.

- Shared Secret Form Line. For each form type that is supported for a taxpayer to inquire about a refund, indicate the XPath of the form line reference. Note that there is an ability to view the schema of the form business object associated with the form type to find the correct XPath.

  Note: Reported vs. Current. Because the configuration is related to information that the taxpayer has reported, consider whether the "current" element (asCurrent) or "reported" element (asReported) is used. If the taxpayer has made calculation mistakes, form rules or users may adjust the current value of the shared secret field. In this case the reported value is what the taxpayer knows and would enter.

- Overpayment Process Amounts. In order to be able to provide details related to the refund, it is necessary to report amounts that may have been used to offset other debt and amounts that are carried forward to future filing periods. The information is defined as elements in overpayment process business object. For each Overpayment Process Type, indicate the XPath values for the Refund Amount, the Offset Amount and the Carry Forward Amount. Note that there is an ability to view the schema of the overpayment process business object associated with the overpayment process type to find the correct XPath.

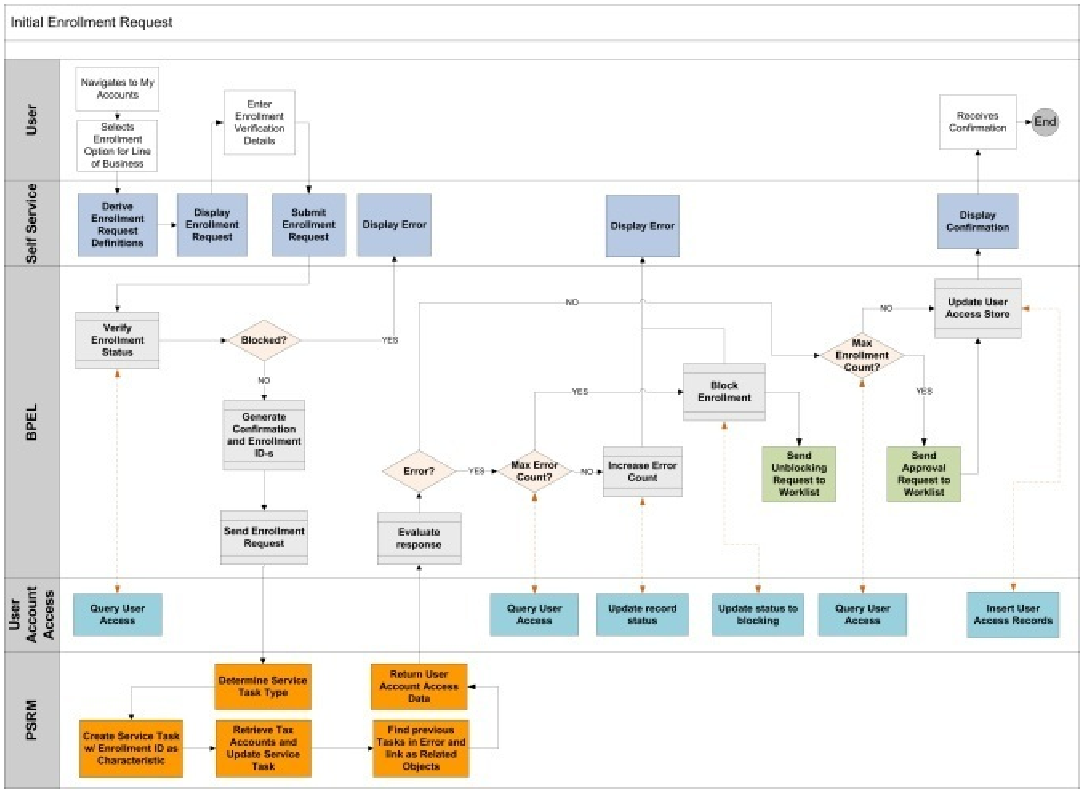
Enrollment Request

This special class of service request supports the ability for a user to request online access to multiple tax accounts and to perform various self-service operations.

One enrollment request is created per line of business.

> **Note:** See the *Enrollment* section for more details about this functionality.

This is a special kind of a service request that defines its own web service, **TSEnrollmentServiceRequest**. The web service requesting the enrollment is processed by the XAI Inbound Service, which invokes the service script **C1-EnrSvcReq**. The appropriate service task type to use must be passed in with the other information received from the self service system.


Initial Enrollment Request
User
Self Service
BPEL
User Account Access
PSRM
Navigates to My Accounts
Selects Enrollment Option for Line of Business
Enter Enrollment Verification Details
Receives Confirmation
End
Derive Enrollment Request Definitions
Display Enrollment Request
Submit Enrollment Request
Display Error
Display Error
Display Confirmation
Verify Enrollment Status
Blocked?
YES
NO
Generate Confirmation and Enrollment ID-s
Send Enrollment Request
Error?
YES
Evaluate response
Max Error Count?
NO
YES
Increase Error Count
Block Enrollment
NO
Max Enrollment Count?
NO
YES
Update User Access Store
Send Unblocking Request to Worklist
Send Approval Request to Worklist
Query User Access
Query User Access
Update record status
Update status to blocking
Query User Access
Insert User Access Records
Determine Service Task Type
Return User Account Access Data
Create Service Task w/ Enrollment ID as Characteristic
Retrieve Tax Accounts and Update Service Task
Find previous Tasks in Error and link as Related Objects

The integration has the following steps:

- The self service application captures information about the taxpayer that is sufficient to identify tax accounts related to this taxpayer within the context of a specific line of business.
- The integration layer generates two unique IDs: **Confirmation ID**, which will be later presented to the self-service user, and **Enrollment ID**, which will be used internally between SOA/BPEL layer and the revenue management system.
- This information is sent to the system, which creates an Enrollment Request service task.
- The task's lifecycle is designed to process the information immediately, with no deferred monitor process, so that the information can be returned to the self service application in real-time.
- The base business object for the Enrollment Request Service Task includes an algorithm, **C1-ENR-INIT**, that reads enrollment service task type configurations and orchestrates the process. It invokes access type-specific logic to verify the taxpayer identity, determine what data this user should be able to access and manage, and populate the results on the response message. The response message contains the list of tax account identifiers (access type+access keys). Each entry also includes the enrollment status, revenue management system, and enrollment ID.

Note: See *Tax Account Access Information* for a detailed explanation of tax account access, and *Configuration Tasks for Enrollment Request* information about enrollment task type configurations.

- The response is sent back to the self-service application immediately.
- Enrollment entries are processed by SOA Composite in the integration layer. They are captured in the user access store table.
- The final response to the self-service application contains confirmation details.

Note that the implementation of the enrollment request creates a service task for the following reasons:

- The creation of a service task with its business object architecture provides many plug-in spots to allow implementations to easily customize the initial enrollment logic.
- The service task type provides a facility for the various configurations related to enrollment handling.
- The enrollment request service task provides an audit tracking for the enrollment activities and is linked to the unique enrollment ID which is shared between the revenue management system(s) and the self service product. It also allows tracking of failed enrollments attempts.
- The enrollment request service task captures the initial list of tax accounts and provides future reference for the implicit enrollment refresh.

  Note: See *Implicit Enrollment Refresh* for more information.

- Creating a service task enables the tax authority to provide the taxpayer with a confirmation ID for possible follow-up . If the taxpayer contacts the tax authority and provides the confirmation ID, the user can view the service task, verify the details, and more easily investigate the situation.

Determining the Tax Accounts Owned by User

The base product supports the enrollment for access type *Tax Role* out-of-box. It is implemented for two lines of business: *Individual* and *Business*.

Access keys used for *Tax Role* are:

PER_ID - taxpayer

TAX_ROLE_ID - tax role

Note: See *Message Processing Overview* chapter for more information about access type and line of business.

The enrollment logic depends on the requested line of business.

- **Individual** enrollment - the logic searches for accounts where the identified taxpayer is listed as either a primary taxpayer or a financially responsible person. Non-cancelled tax roles linked to these accounts are returned.
- **Business** enrollment - the logic reads the taxpayer instructions on the service task type and determines the relationships between the requestor and the business. It then determines the business person and finds all accounts where this person is either a primary taxpayer or a financially responsible person. The non-cancelled tax roles linked to these accounts are returned.

For example, the *Taxpayer Instruction* on the service task type lists *Corporate Officer* person-to-person relationship type means that the taxpayer(person) requesting an access to the business tax account should be linked to this business (person) as a corporate officer.

An algorithm plugged into the enrollment service task business object as an enter plug-in on the In Progress state is responsible for determining the tax accounts. Refer to the description of the algorithm for more information about the base logic provided.

See *Configuration Tasks for Enrollment Request* for information about configuring the system to support enrollment requests.

Configuration Tasks for Enrollment Request

The following sections list the configuration tasks required to implement the base taxpayer enrollment request functionality.

Characteristic Type

If your implementation captures the taxpayer's birth date on the person record and would like that to be part of the taxpayer identification for enrollment service requests, create a characteristic type for Birth Date. It should be defined as ad-hoc and should reference Person as the characteristic entity.

Notes:

- It is recommended to use the same characteristic type as the one defined for *Taxpayer Identification Service Request.*
- The product does not provide any functionality for populating the birth date characteristic on the person.

Service Task Type

Create a service task type for the enrollment request service task for each line of business your implementation will support. Using the business object **User Enrollment Self Service Request Type**:

- Select the line of business from the drop-down list.
- The related transaction BO should be set as follows:
  - Line of Business ***Business*** - C1-BusinessEnrollmentRequest.
  - Line of Business ***Individuals*** - C1-IndividualEnrollmentRequest
- Service task class should be set to Service Request.
- To Do Type should be set to a standard error To Do type to use when transitioning to error. The base product To Do Type Self Service Task Issues (C1-SSTTD) is provided for this purpose.
- To Do Role should be configured to an appropriate default To Do role for the above To Do type. This is optional. If no value is provided the default role for the To Do type is used.
- Configure the Confirmation Header Message Category and Message Number and the Error Header Message Category and Message Number to use for communication to the self service user.

  Note: See the Messages section of the *General Configuration Tasks* for more information.

- The service request fields mapping is used when creating the target service task to correctly populate the requestField list from the list of fields passed in on the web service request XML. For the base transaction business object, the following fields are expected:

Sequence	Transaction BO XPath
1	requestData/idType
2	requestData/idValue
3	requestData/birthDate

- **Enrollment Instructions**:
  - Char Type for Date of Birth - Specify the characteristic type defined above

- **Name Type** - Specify what type of taxpayer's name should be displayed on enrollment summary. This configuration is used by enrollment summary retrievers.

  Note: See *Enrollment Summary* for more information.

- **Enrollment status** - Specify the default status of the user access records. Permissible values are *Approved*, *Pending* and *On Hold*.

Business

- **Taxpayer Instructions:** List person-to-person relationships that may link a user requesting an access to the tax account owner. In other words, define who may be granted an online access to the account: business owner, corporate officer, legislative advisor. List all relationships that should be considered by enrollment logic.
- **Access Type Processors:**

  If your implementation wish to support user account access on the tax role level provided in the base product, configure this section as follows:

  - Select an access type Tax Role.
  - Specify the business service **C1-EnrolIndividualAccntTaxRole** as a User Enrollment Retriever.
  - Specify the business service **C1-GetBusinessTaxRoleSummary** as an Access Type Summary Retriever.

Individuals

- **Taxpayer Instructions:** if your implementation wish to utilize User Summary Retriever included in the base product, skip this section. The individual tax account determination logic is not considering these instructions.
- **Access Type Processors:**

  If your implementation wish to support user account access on the tax role level provided in the base product configure this section as follows:

  - Select an access type Tax Role.
  - Specify the business service **C1-EnrolIndividualAccntTaxRole** as a User Enrollment Retriever.
  - Specify the business service **C1-GetIndividualTaxRoleSummary** as an Access Type Summary Retriever.

  Note: Service Task Type Mapping. The service task type defined here must be configured in the service task domain value mapping in the SOA Composer application. Refer to the self service product documentation for details.

Implementing a New Service Request

Your implementation may wish to provide support for additional service requests for your taxpayers. The following information outlines the steps required in the product to support a new service request.

Taxpayer Identification

First decide if the taxpayer must provide identification information as part of the service request. If so,

- Should the system validate the taxpayer before continuing with the actual request? In this case an initial web service call must be sent to the system with appropriate information to validate the taxpayer.

- Can the identification information be provided with the other service request details so that the taxpayer identification can be done with the service request processing?

If the taxpayer identification is done as a separate step, refer to *Configuration Tasks for Taxpayer Service Request* for the base taxpayer identification business object provided with the product. That may be used or a new one based on appropriate business rules may be introduced with this one as a sample.

Business Object

Each service request must have its own Service Task business object. This business object defines the information required for this type of request and defines the algorithms that validate and process the request. The product provides several business objects that may be used as a parent business object for a new service request.

- The Standard Deferred Self Service Task business object (**C1-StandardDeferredSSTask**) supports the ability to defer the full processing of the request to a subsequent step. Defining a service task business object as a child of this business object is recommended if any of the following are true:
  - A user needs to review the taxpayer's request.
  - The request is high volume such that processing the request real-time for many taxpayers may cause a lot of system traffic.
  - The logic required to process the request is such that it doesn't make sense for the taxpayer to wait until the processing is complete.
- The Standard Real Time Self Service Task business object (**C1-StandardRealTimeSSTask**) supports immediate processing of the service request to give a response to the taxpayer real time.
- The Standard Scheduled Reminder Self Service Task business object (C1-ScheduledReminderSSTask) supports the repeatable creation of the reminder, maintaining the reminder task in an active status and expire it at the specific date. Defining a service task business object as a child of this business object is recommended if any of the following are true:
  - A periodic reminder should be issued.
  - The date of the reminder may be derived from the request data or calculated based on some conditions.

Create the new business object defining the appropriate parent business object and navigate to the schema tab. Include the parent BO's schema. Define the additional elements required for this business object's functionality. Refer to the Tax Clearance Certificate business object for an example.

The business object must also include the appropriate algorithms to satisfy the request.

Service Task Type

Create a service task type for the new service request with the appropriate service task type business object; base product provides business objects for **Taxpayer Service Request Self Service Task Type**, **Generic Create To Do Request Task Type** and **Scheduled Reminder Request Task Type**.

- The related transaction BO should be set to the business object created above.
- Service task class should be set to **Service Request**.
- To Do Type should be set to a standard error To Do type to use when transitioning to error. The base product To Do Type **Self Service Task Issues (C1-SSTTD)** is provided for this purpose.
- To Do Role should be configured to an appropriate default To Do role for the above To Do type. This is optional. If no value is provided the default role for the To Do type is used.
- Configure the Confirmation Header Message Category and Message Number and the Error Header Message Category and Message Number to use for communication to the self service user.

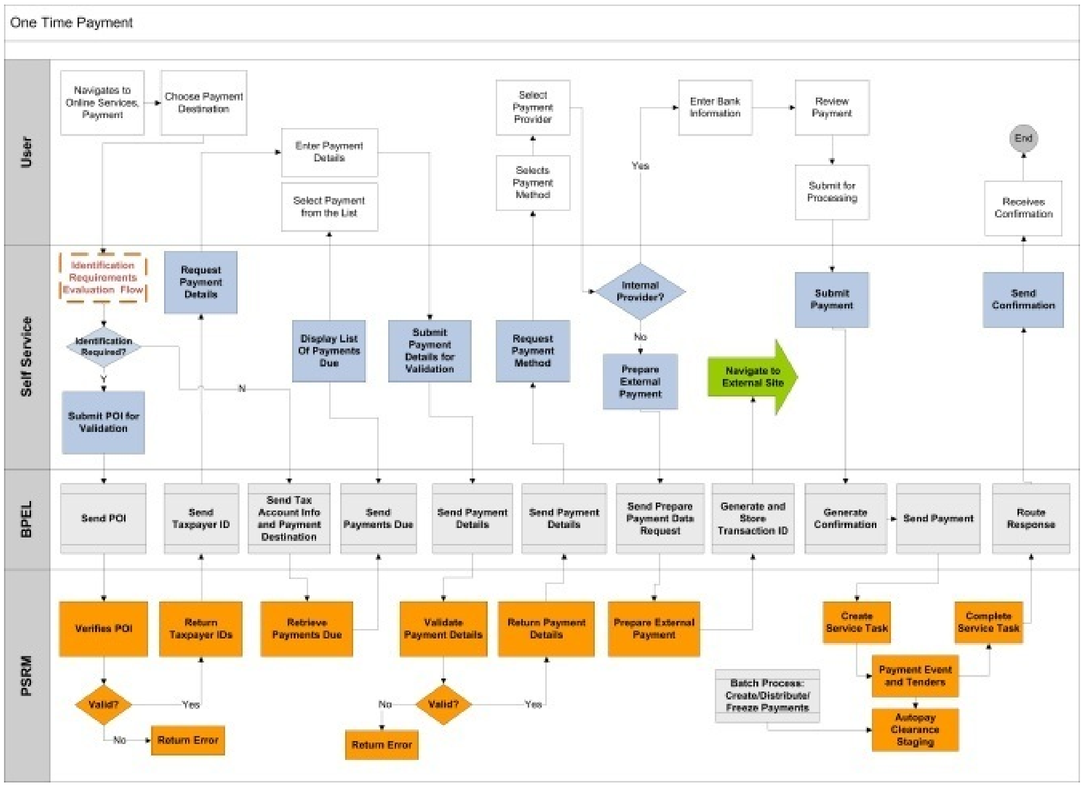
Fastpath: Refer to the Messages section of the *General Configuration Tasks* for more information.

- The service request fields mapping is used when creating the target service task to correctly populate the request data list from the list of fields passed in on the web service. Define the mappings appropriate for this service request as needed.
- Add request-specific configurations where needed.

Fastpath: Additional Configuration is required in the self service product and in the integration layer for new service requests. Refer to the self service product documentation for more information.

Online Payment

In this release, the integration supports payment by a taxpayer using a bank or credit card number. The following diagram illustrates an expected process flow.



The integration has the following steps:

- Initial processing works as follows:
  - **Casual user**. The self service application captures information about the taxpayer and sends a web service request to this product to validate the taxpayer (proof of identity, or POI).
  - **Enrolled user**. The identification is not performed.

  Fastpath: Refer to the *User Identity Verification* topic for more information.

- Once identified, the casual user indicates what the payment is for. This is referred to as the target of the payment or the payment destination or the payment details. For example, is this a payment for a pay plan scheduled payment? If so, the taxpayer must provide the pay plan identifier. Once the taxpayer indicates the payment target details, another web service request is sent to this product to verify the details.

- When the enrolled user is browsing a specific tax account it becomes possible to retrieve payments that are currently due and present user with the pre-populated list so the user will choose what to pay rather than enter payment details manually.
- Next, the taxpayer indicates the method of payment:
  - If the method is via a credit card and the implementation uses an external payment vendor, the self service application sends a web service request to this product to retrieve any additional information about the taxpayer that may need to be sent to the external vendor (such as the taxpayer's address) and to determine if the external vendor's fee should be passed on to the taxpayer. At that point the taxpayer is taken to the external vendor's website to further process the payment details. Once that step is complete a final web service request is sent to this product to process the payment.
  - If the method is via a bank account or if the implementation does not use an external payment vendor for credit card payments, the self service application sends a web service to this product to create the payment. The standard Auto Pay logic in the product is used to process the payment.
- If the external vendor supports payment reconciliation, the subsequent reconciliation report is processed by the product.

The topics below describe more information about the provided functionality, followed by configuration tasks.

Identifying the Taxpayer

In this release the integration supports payments by an 'unregistered' taxpayer. The taxpayer does not need to log in using a user name and password in order to submit a payment. In addition, the product supports a person paying on behalf of another taxpayer.

The taxpayer must provide enough information to be identified by the system prior to proceeding. A special service task business object is provided for identifying a taxpayer for one-time payments (**C1-TaxpayerIdentifySSTask**).

Fastpath:
Refer to *Configuration Tasks for Online Payment* for more information.

Payment Destination

When creating a payment, the product supports using distribution rules to determine where the payment should be targeted. The integration provides out of the box support for the following payment destinations.

- **Pay a filing period**. This method allows the taxpayer to provide their taxpayer ID, a tax type and a filing period to direct the payment to.
- **Pay a pay plan**. This method allows the taxpayer to provide a reference to a pay plan to pay one or more of the scheduled payments.
- **Pay a collection notice**. This method assumes that a collection notice has been issued from a customer contact in the system and that the customer contact ID has been provided to the taxpayer as an identifier.
- **Pay a Form (DLN Only)**. This method assumes that a tax or business registration form was processed by the system and the Document Locator Number has been provided to the taxpayer.
- **Pay an Obligation**. This method allows taxpayer to provide the exact ID of the obligation and pay obligation's balance.
- **Pay a Tax Role Balance**. This method allows taxpayer to provide the exact tax role ID and pay an outstanding tax role balance.
- **Pay a Tax Form**. This method allows the taxpayer to provide a DLN of the form that is not necessarily exists in the system yet. The assumption is that this method would be used for payments immediately following the on-line form submission for the scenario where the form submission web service request is queued by the integration while the payment request is processed immediately.

The business object provided by the product for processing one time payments has been designed to support different payment destinations. To do this, the business object must support receiving different types of information (for example, pay plan ID or collection notice ID) and use plug-ins that are specific to each payment destination that knows the type of data expected, how to use that data to identify the correct account and obligation(s) and process the payment successfully. At a high level it works as follows:

- The web service requests sent by the self service application to validate the payment target and to process the payment include an indication of the payment destination and a field/value pair collection to hold the values that represent the 'target' for the payment.
- The self service master configuration record includes a mapping between the payment destination and a self service task type.
- The service task type for each unique payment destination references a "processor" service, which can be a service script or a business service. This service should be specific for a given payment destination and expects the data appropriate for it. For example, the process to "pay a pay plan" expects the service task's destination details to specify a pay plan ID.
- The service task type includes a field mapping section. This mapping indicates which field value in the destination details list to populate in which target XPath in the "processor" service.
- The web service called to validate the destination details provided by the taxpayer invokes the "processor" to ensure that the data can be found.
- When processing the payment, the service task that processes payments includes an algorithm that uses the configuration on the task type to call the "processor" service, passing the details of the payment target.

Fastpath: Refer to *Configuration Tasks for Online Payment* for information about configuring the system to support these payment destinations. Refer to *Implementing New Payment Destinations* for information about defining additional payment destinations for your implementation.

Account Verification Service

The processor is used to verify if the input account is valid for the entity identified by the current access information. If the account is invalid it returns an error.

When creating a new custom Processor, use any of the base business services or service scripts provided as an example of the type of functionality needed. All base components provided begin with **C1-OTPVAc%**. The processor may be implemented as a service script or a business service.

The custom service, whether it be a service script or a business service should include the data area **C1-SelfServiceKeys** along with an **<accountId>** element

Retrieving Payments Due

The web service used to retrieve the list of outstanding payments is processed by the XAI Inbound Service TSRetrievePaymentsDue, which invokes the service script **C1-RtPymtDue**.

The web service request contains:

- Access info: access type and access keys – Identifies the enrollment unit, for example Tax Role, Account or others
- Payment destination defines what type of payment is expected; for example, collection notice, obligation, etc.

The service script determines the service task type based on the payment destination (via master configuration). It then calls the retrieve payments due processor. The processor should return a collection of the prospective payments. Each entry contains:

- Payment amount due.
- Payment destination fields collection.
- Parameters collection used by self-service system to display the formatted info about this payment. Parameters are injected into the message defined in the self-service Payment Destination admin configuration.

The parameters for the obligation info display are obligation type and payment due date.

Note: Integration Mapping should be completed for proper display of obligation information.

Validating the Payment Destination

The web service used to validate the payment destination information entered by the taxpayer is processed by the XAI Inbound Service **TSPrepareExtPaymentData**, which invokes the service script **C1-PrExPyDta** with an action of **VALIDATEONLY**. The service script determines the service task type based on the payment destination (via master configuration). It then calls the payment destination details processor with an action of "validate only". The processor should return an error if the information provided by the taxpayer does not identify an appropriate related object for the payment destination.

If the enrolled user is making a payment, the service script calls an additional processor that checks if the account determined during the destination details validation is relevant for the tax account in context (access type+ access keys).

Creating the Appropriate Service Task

The web service request to create a payment is processed by the XAI Inbound Service **TSOneTimePayment**, which invokes the service script **C1-PyUnrgUsr**. The service script is responsible for creating a Service Task of the correct service task type based on the payment destination.

Processing Auto Pay Payments In the System

When a taxpayer provides bank account information for payment in self service, the payment detail is sent to this system and the integration with the customer's bank is handled through standard auto pay processing. This method may also be used to process credit cards. However, it's expected that an integration with an external vendor will most often be used for that. Refer to the section below for more information on external payment vendors.

Requiring an Auto Payment Agreement

Many tax authorities require taxpayers to sign an automatic payment agreement prior to allowing them to submit tax payments online. The product supports checking for a reference to an ACH agreement that is defined as a characteristic linked to the taxpayer's record.

Payment Processing Lifecycle

The one-time payment service task provided by the base product includes the following basic lifecycle states:

- **Pending**. When the web service request is processed, a service task is created in pending status. Initial validation is performed, including verifying that the information provided for the payment destination and determining the appropriate account to associate with the payment. If any issues are found, the service task is not stored and an error is returned. Once the record is validated, no further processing occurs at this time. When the service task monitor is run all pending records are further processed.
- **In Progress**. When transitioning to this status, the payment is processed. The data related to the payment destination is mapped to appropriate payment event distribution details and the payment is created. If any errors are encountered, the service task transitions to **Error**. If the payment was processed through an external vendor that requires reconciliation, it transitions to **Pending Reconciliation**. Otherwise, it transitions to **Complete**.
- **Error**. Any errors detected in payment processing or during reconciliation processing causes the service task to transition to this state and creates a To Do entry. A user must review and resolve the problem.
- **Error Resolved**. When a user resolves the error reported the service task can be transitioned to this state. Note that the assumption is that the user has manually corrected the error. This state does not cause any logic to occur.
- **Pending Reconciliation**. When a payment is made through an external vendor that requires reconciliation, the service task transitions to this state and waits for the reconciliation process. Refer to the external payment vendor section below for more information about reconciliation. Note that this state is configured with time out logic. The payment vendor can

be configured with a number of days to wait for the reconciliation report. If that number of days passes, a To Do entry is created using an appropriate To Do type defined for the payment vendor.

- **Complete**. Service tasks transition to this state when the payment is fully processed.

External Payment Vendor

Many websites integrate with an external payment vender to process credit cards real-time. In this case, the following functionality is provided:

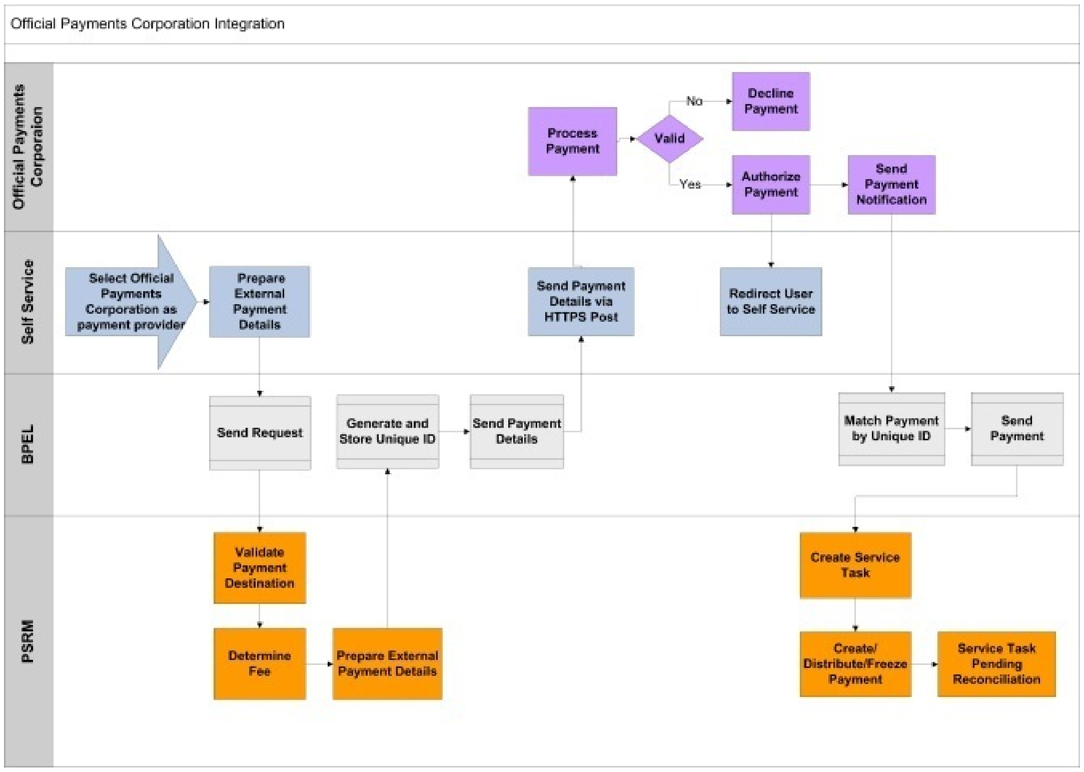
- Preparing External Payment Data. After the standard steps of identifying the taxpayer and validating the payment destination, when a taxpayer indicates payment using a credit card, the self service system sends a web service request to this product to determine whether a fee is applicable and to return additional payment data.
- Payment processing. Once the credit card information is processed by the external vendor, a web service request with the payment information is sent to this system. This causes an appropriate service task to be created to process the payment and apply it to the taxpayer's balance appropriately.
- Reconciliation. External vendors may supply reconciliation information for all the payments that were processed in a given time period. The product supports the ability to receive the reconciliation information, compare it against our records and highlight any discrepancies.

Note: Integration with Official Payments Corporation (OPC). The product provides functionality to integrate with OPC as an external vendor. The logic provided in the base product is based on that vendor's requirements.

Refer to the self service documentation for more information about configuring the system to work with external vendors.

The following topics provide additional detail about the integration functionality provided.

The following diagram illustrates the prepare external payment data and payment processing components for the integration with OPC.


Official Payments Corporation Integration
Official Payments Corporaion
Self Service
BPEL
PSRM
Select Official Payments Corporation as payment provider
Prepare External Payment Details
Send Request
Validate Payment Destination
Determine Fee
Prepare External Payment Details
Generate and Store Unique ID
Send Payment Details
Send Payment Details via HTTPS Post
Process Payment
Valid
No
Yes
Decline Payment
Authorize Payment
Send Payment Notification
Redirect User to Self Service
Match Payment by Unique ID
Send Payment
Create Service Task
Create/ Distribute/Freeze Payment
Service Task Pending Reconciliation

Prepare External Payment Data

Once the taxpayer has indicated that payment will be made by credit card, the self service application sends a web service request to determine the fee requirement and to retrieve detail about the taxpayer to provide to the external vendor. These are processed by the XAI Inbound Service **TSPrepareExtPaymentData**, with an action of **PREPARE**, which invokes the service script **C1-PrExPyDta**. This script does the following:

- It uses the payment destination to determine the appropriate service task type (via the master configuration) and uses this information to determine the fee requirements. External vendors often charge fees for using their service. The tax authority may to choose to pass the fee onto the taxpayer or to absorb the cost of the fee. The product supplies configuration to allow the tax authority to define for each service task type whether taxpayers should be charged the fee based on the person's person type.

  Note: Refer to the Service Task Types section of *Configuration Tasks for Online Payments* for more information.

- It retrieves additional information about the taxpayer, if required by the external vendor. It looks for a Prepare Data service script linked to the external vendor's record. If one is defined it is called. Refer to *Configuration Tasks for Payment Vendors* for more information.

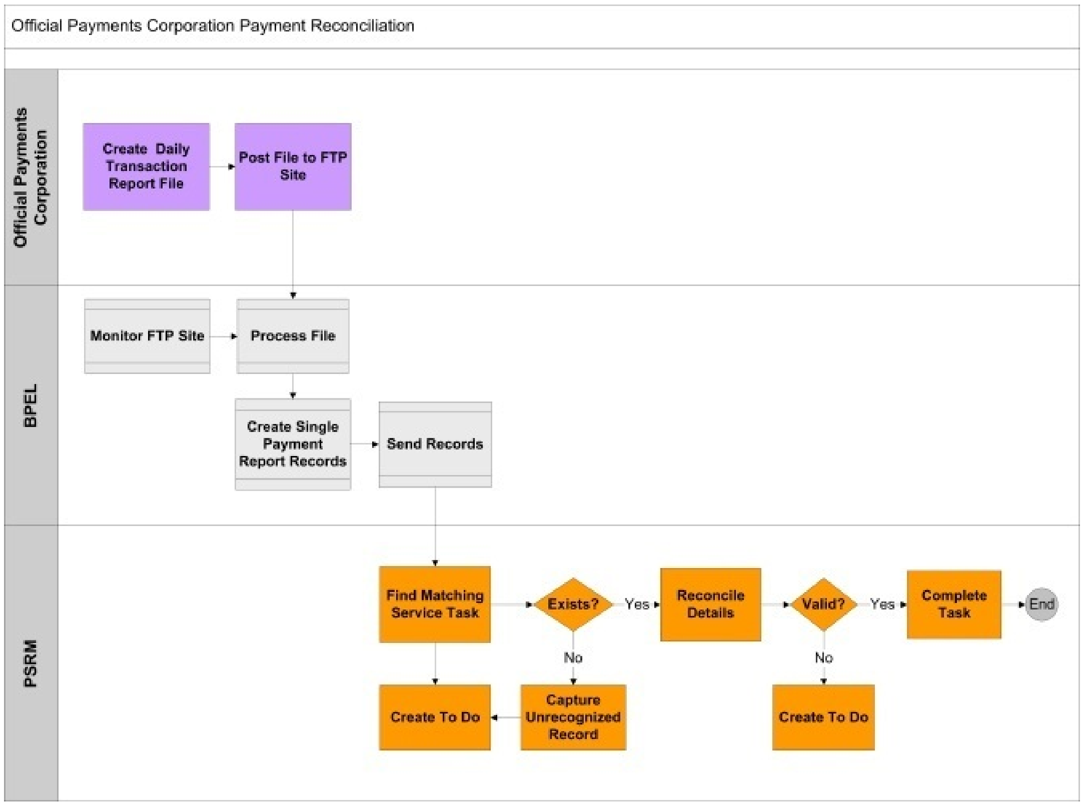
Payment Processing

For the most part, the payment processing for these payments is analogous to the processing done for bank payments. The tender type used should be one that is not configured to integrate bank details. And an external reference to the payment vendor's record should be recorded with the service task using a characteristic. This allows the reconciliation process to identify the record.

The product supports capturing additional information as defined by the requirements for the external vendor. The data is stored in a "raw" element on the service task called externalPaymentData. The structure of the information captured must be defined using the External Payment Data Area linked to the payment vendor lookup.

Reconciliation

The following diagram illustrates the reconciliation component for the integration with OPC.


Official Payments Corporation Payment Reconciliation
Official Payments Corporation
Create Daily Transaction Report File
Post File to FTP Site
BPEL
Monitor FTP Site
Process File
Create Single Payment Report Records
Send Records
PSRM
Find Matching Service Task
Exists?
Yes
No
Reconcile Details
Valid?
Yes
No
Complete Task
End
Create To Do
Capture Unrecognized Record
Create To Do

Once the service task successfully creates the payment, it checks whether reconciliation with the payment vendor is applicable and if so, it transitions to the **Pending Reconciliation** state to wait for the reconciliation details.

When the reconciliation details are processed, individual web service calls are sent to the system for each payment. These are processed by the XAI Inbound Service **TSProcessExtPayReportRecord**, which invokes the service script **C1-PrcPyRpRc**. This script uses the external reference to find an existing payment service task in the **Pending Reconciliation** state with this reference linked as a characteristic.

- If one is found, it is updated with a snapshot of the payment report from the vendor. The payment report data is stored on the service task in a "raw" element called externalPaymentReport. The structure of the information captured must be defined using the External Payment Report Data Area linked to the payment vendor lookup. In addition, it is marked as ready to reconcile. The next time the reconciliation monitor runs the service task's reconciliation algorithm is executed. The algorithm is configured to call logic that is specific for the payment vendor. It calls the Payment Reconciliation Script defined on the payment vendor lookup. If there are any issues it transitions to **Error**, otherwise it transitions to **Complete**.
- If one is not found, the system creates a new service task record using a special task type configured for logging an unrecognized external payment in the payment report. The service task BO provided by the product for this scenario **C1-UnrecognizedExtPymtRptTsk** provides a simple lifecycle that creates the record in the **Pending** state. The next time the service task deferred monitor runs, the record creates a To Do entry for a user to research and resolve the issue and sets this record to **Complete**.

Fastpath:
Refer to *Configuration Tasks for Payment Vendors* for more information.

Configuration Tasks for Online Payments

The following sections list the configuration tasks needed to implement the online payment functionality.

Standard Payment and Automatic Payment Options

When the one-time payment service task processes the payments it relies on standard payment logic in the product. Refer to the payment configuration documentation for configuration options required for general payment processing as well as configuration options for automatic payment processing.

> **Fastpath:** Refer to the Defining Financial Transaction Options > Managing Payment Setup and Automatic Payment Options in the administration guide for more information.

Obligation Type for Excess Credit

The base product payment distribution rule algorithms that create payments support the ability to direct excess credit to a special obligation rather than crediting the obligation used to levy the taxes. If your implementation uses the base product algorithms, define an obligation type to use when directing excess credit to a special obligation for the account.

Suspense Obligation

In some cases when an account cannot be found for directing a payment, a payment is created in suspense, meaning that it is created for a suspense obligation. If your implementation does not already have a suspense obligation set up, create one.

Match Types

Define a match type for Obligation. This is used by the Pay a Pay Plan distribution rule.

Algorithms

Create a Distribution Rule - Create Payment algorithm for each supported payment destination.

- **Pay a Filing Period**. If your implementation supports the ability to direct a payment to a filing period, define an algorithm for the algorithm type **C1-PAYFRM**. In the parameters define the division / obligation type for the excess credit obligation along with the suspense obligation. Finally, define characteristic types for each of the payment details that are required by the algorithm. Note that the product supplies characteristic types that may be used for each one.
  - C1-DLN for document locator number. Note that this is not expected to be supplied by a taxpayer for web self service payments, but the parameter is required for the algorithm.
  - C1-TXPID for taxpayer ID
  - C1-TAXTY for tax type
  - C1-FLNGP for filing period
- **Pay a Collection Notice**. If your implementation supports the ability to direct a payment to a collection notice, define an algorithm for the algorithm type **C1-DR-PAYCC**. In the parameters define the division / obligation type for the excess credit obligation. In addition, define the characteristic type that the overdue process uses to log the creation of the customer contact. This allows the algorithm to determine the related overdue process to find the covered obligations.
- **Pay a Pay Plan**. If your implementation supports the ability to direct a payment to a pay plan, define an algorithm for the algorithm type **C1-DR-PAYPP**. In the parameters define the division / obligation type for the excess credit obligation. In addition, define the match type for referencing the pay plan's obligation.
- **Pay a Tax Role Balance**. If your implementation supports the ability to direct a payment to a tax role balance, define an algorithm for the algorithm type C1-DR-PAYTXR. In the parameters define the division / obligation type for the excess credit obligation.
- **Pay an Obligation**. If your implementation supports the ability to direct a payment to an obligation, you may use an algorithm C1-DR-PAYOBL delivered with the product.

Distribution Rules

Create a distribution rule for each supported payment destination.

- **Pay a Filing Period**. If your implementation supports the ability to direct a payment to a filing period, multiple distribution rules must be created, one for Taxpayer ID, one for Tax Type and one for Filing Period. Each should reference the appropriate characteristic type. Refer to the create payment algorithm above for details of the characteristic types to use. Note the following points related to configuring these distribution rules:
  - The payment distribution processing expects that the create payment algorithm is only linked to the last distribution rule.
  - The payment distribution processing expects that the Determine Tender Account is linked to all the distribution rules.
  - The payment details processor business service used by the payment service task may expect the definition of the distribution rules to be in a certain order. Refer to the business service description for **C1-PaymentDestinationTxTyFilPd** for more information.
- **Pay a Collection Notice**. If your implementation supports the ability to direct a payment to a collection notice, create a distribution rule referencing the characteristic type **C1-CUSCN** Customer Contact. It should also reference the Create Payment algorithm created above for paying a collection notice.
- **Pay a Pay Plan**. If your implementation supports the ability to direct a payment to a pay plan, create a distribution rule referencing the characteristic type **C1-PAYPL** Pay Plan. It should also reference the Create Payment algorithm created above for paying a pay plan.
- **Pay a Form (DLN Only)**. If your implementation supports the ability to direct a payment to a form using DLN only, create a distribution rule referencing the characteristic type C1-DLN GDocument Locator. It should also reference the Create Payment Algorithm created above for paying a filing period.
- **Pay an Obligation**. If your implementation supports the ability to direct a payment to an obligation, create a distribution rule referencing the characteristic type C1-OBLIG GObligation. It should also reference the Create Payment algorithm created above for paying an obligation.
- **Pay a Tax Role Balance**. If your implementation supports the ability to direct a payment to a tax role balance, create a distribution rule referencing the characteristic type C1-TAXRL Tax Role. It should also reference the Create Payment Algorithm created above for paying a tax role balance.

Business Object

Decide whether or not an email confirmation should be sent to the taxpayer once the payment is processed. If so, add an Enter plug-in to the In Progress state after the payment creation algorithm or perhaps the Complete state to generate the confirmation email.

> **Fastpath:** Refer to the algorithm section of the *General Configuration Tasks* for more information about configuring the confirmation algorithm.

Service Task Types

Taxpayer Identification Task Type

Define a task type to use for taxpayer identification with the business object **Taxpayer Service Request Self Service Task Type**.

- The Related Transaction BO should be set to **C1-TaxpayerIdentifySSTask**.
- Service Task Class should be set to **Taxpayer Identification**.
- To Do Type should be set to a standard error To Do type to use when transitioning to error. The base product To Do Type **Self Service Task Issues (C1-SSTTD)** is provided for this purpose. Note that the algorithm plugged in on the error

state for the base business object checks that there is not already a To Do for the record before creating one of this type. This allows algorithms that detect an error to also create a To Do with a specific type if desired.

- To Do Role should be configured to an appropriate default To Do role for the above To Do type. This is optional. If no value is provided the default role for the To Do type is used.
- Configure the Confirmation Header Message Category and Message Number and the Error Header Message Category and Message Number to use for communication to the self service user.

Fastpath: Refer to the Messages section of the *General Configuration Tasks* for more information.

- The service request fields mapping is used when creating the target service task to correctly populate the requestField list from the list of fields passed in on the web service request XML. For the base transaction business object, the following fields are expected:

Sequence	Transaction BO XPath
1	requestData/legalName
2	requestData/idType
3	requestData/personIdNumber
4	requestData/emailAdd
5	requestData/addressLine1
6	requestData/addressLine2
7	requestData/city
8	requestData/county
9	requestData/state
10	requestData/postal
11	requestData/onBehalfOfFlag
12	requestData/secondaryLegalName
13	requestData/secondaryIdType
14	requestData/secondaryPersonIdNumber
15	requestData/secondaryEmailAdd

Fastpath: Service Task Type Mapping. The service task type defined here must be configured in the service task domain value mapping in the SOA Composer application. Refer to the self service product documentation for details.

Payment Destination Task Types

Create a service task type for each payment destination with the business object **One-Time Payment Self Service Task Type**. Each service task configured should include the following common configuration.

- The Related Transaction BO should be set to **C1-StandardOneTimePaySSTask**.
- Service Task Class should be set to **Self Service Payment**.
- To Do Type should be set to a standard error To Do type to use when transitioning to error. The base product To Do Type **Self Service Task Issues (C1-SSTTD)** is provided for this purpose. Note that the algorithm plugged in on the error state for the base business object checks that there is not already a To Do for the record before creating one of this type. This allows algorithms that detect an error to also create a To Do with a specific type if desired.
- To Do Role should be configured to an appropriate default To Do role for the above To Do type. This is optional. If no value is provided the default role for the To Do type is used.

- Configure the Confirmation Header Message Category and Message Number and the Error Header Message Category and Message Number to use for communication to the self service user.

  Fastpath: Refer to the Messages section of the *General Configuration Tasks* for more information.

The following configuration differs based on the payment destination:

- Payment Distribution Rules. Link the appropriate distribution rule(s) created above for the payment destination represented by this service task type.
- Link an appropriate Processor business service or service script that knows how to receive the detail related to the payment destination and create the payment event distribution details correctly. The base product provides one Processor business service for each supported payment destination. Using the search look for business services that begin with **C1-PaymentDestination%**.
- Specify if taxpayer should exist in the system in order to process the payment. Typically this should be set to **Y**es ("Y") as the taxpayer record is necessary for account and other payment details verification. An example of the exception from this rule is a payment for a form by DLN. When payment is submitted immediately following online form filing the system should be able to accept payment from the self-service user who is a first-time filer. In this scenario the taxpayer record is created during form posting and that may actually happen after payment submission.
- Field mappings. For each payment destination detail captured on the service task, indicate the target XPath value in the above processor.

  Note: **Special note on payment destination Tax Form (C1OF)**: This payment destination represents a payment for a form that was submitted online but has not necessarily reached the system at time of payment. For a service task type for this payment destination use processor ***C1-PaymentDestOnlineFormFiling*** and the distribution rule **Pay a Form (DLN only)** created above.

Configure the appropriate Processors for access type-specific logic. For each access type:

- **Account Verification**. This processor verifies whether account determined based on payment destination details is valid for the service request access information. This verification is optional and should be performed according to the business requirements. Account verification is needed if the payment has to be made within the context of the current access entity.

  Consider the following example: a self-service user is currently browsing a tax role and attempts to pay a collection notice. Configure account verification for access type **Tax Role** if you want user to be able to pay only for those collection notices associated with the tax role in context. Using the search look for business services or service scripts that begin with **C1-OTPVAc%**.
- **Payments Due**. This processor retrieves all pending items belonging to a payment destination within the context of the current access entity. For example, for payment destination Pay an Obligation and access type **Tax Role** it may retrieve all obligations with an outstanding balance. Using the search, look up business services or service scripts that begin with **C1-RetPaymentsDue%**.

  The examples of payments due retrievers provided out of the box:

  - ***Pay an Obligation*** and the access type ***Tax Role*** . It retrieves all the obligations with an outstanding balance linked to the tax role identified by access key 2). If your implementation supports this payment destination, specify **C1-RetPymtsDueObligationTxRole**.
  - ***Pay a Tax Role Balance*** and the access type ***Tax Role*** . It retrieves a single entry for an outstanding balance for the tax role identified by access key 2). If your implementation supports this payment destination, specify **C1-RetPymtsDueTaxRoleBalance**.

Finally, if external vendors are supported, indicate whether or not fees are appropriate based on the taxpayer type.

Fastpath: Refer to *Implementing A Fee Requirement Script* for information.

Note:
The product supports two distinct methods of paying a form by DLN:

- Payment immediately following the online form filing, supporting first-time filers. For this task type, specify that the taxpayer does not have to exist in the system.
- Payment is made by a registered taxpayer for an existing form.

Master Configuration

The self service master configuration record includes several settings required for payment related processing.

- Char Type for Overdue Process Log. If your implementation supports the Pay a Collection Notice payment destination, define the characteristic type that is used to link the customer contact to the overdue process. The base product provides a value that may be used.
- Refer to the ACH Agreement Validation section below for information about the Char Type for ACH Agreement Verification.
- Tender Types. Indicate the tender types that are valid for paying online.
- Supported Payment Destinations. For each supported payment destination (as defined in the **C1_PAYMENT_TARGET_FLG** lookup), indicate the corresponding Service Task Type.
- Define the customer contact types that represent the collection notices that a taxpayer may pay online using the Pay a Collection Notice payment destination.

ACH Agreement Validation

If your implementation requires a pre-signed agreement from the taxpayer when submitting a payment through their bank account, define a characteristic type on the taxpayer, the following configuration is needed:

- Create an appropriate characteristic type for **ACH Agreement**. Define a valid characteristic entity of **Person**.
- Update the master configuration to indicate this characteristic type.
- Update the business object **C1-StandardOneTimePaySSTask** to plug in the validation algorithm to check for the existence of this characteristic. Navigate to the **Lifecycle** tab and expand the configuration for the **Pending** state. Add an entry in the Algorithms collection with a System Event of **Enter**, an appropriate sequence and the algorithm **C1-CHKACHAGR**.

Note: The product does not provide any functionality for populating the characteristic on the taxpayer record. This functionality is custom and must be provided by your implementation.

Integration Mapping

Obligation types used by the product should have corresponding values defined in the self service product.

Your implementation may decide to use exactly the same values in both products or configure the mapping using domain value mapping in the SOA Composer application

Fastpath: Refer to the self service product documentation for details.

Configuration Tasks for Payment Vendors

The following sections list the additional configuration tasks needed to support an external payment vendor.

External Payment Report Task Type

Create a service task type to use when a payment in the reconciliation report for an external vendor is not found.

- The service task type BO should be **C1-StandardSelfServiceTaskType**
- The related transaction BO should be set to **C1-UnrecognizedExtPymtRptTsk**.
- To Do Type is not applicable. The service task BO has an algorithm that creates a To Do based using the unrecognized payment To Do Type on the payment vendor lookup.
- To Do Role is not applicable
- Confirmation Header Message Category and Message Number and Error Header Message Category and Message Number are not applicable.

External Vendor Extendable Lookup

Create a payment vendor extendable lookup record for the external vendor that is used by the self service application.

- If additional information is required by the external vendor to process the payment (such as the taxpayer's address), indicate an appropriate **Prepare Payment Data Script** that populates the data. The script should use the data area **C1-ExternalPaymentDetails** as its schema. The data area defines a paymentDetails list, which is a collection of field name and value pairs, to capture the additional information.

  Note: Integration with Official Payments. The product provides a service script **C1-BldExPyDt** that builds the additional data required for integration with Official Payments.

- If additional information is included in the payment processed by the external vendor, indicate the **External Payment Data Area** that describes the structure of the data. This information is captured in the service task created to process the payment in the raw element externalPaymentData.

  Note: Integration with Official Payments. The product provides the data area **C1-OPCExtPaymentData** that supports the additional data sent with payments made via integration with Official Payments.

If the vendor supports a reconciliation report, check the **Reconciliation Required** checkbox and fill in the remaining detail.

- Indicate the **External Payment Report Task Type** created above.
- Define the **External Payment Report Data Area** that describes the structure of the data supplied with the reconciliation report.

  Note: Integration with Official Payments. The product provides the data area **C1-OPCExtPaymentReportData** that defines the data provided in the reconciliation report from Official Payments.

- Indicate the **Number Of Days To Wait For Timeout** appropriate for your implementation.
- Define the **Payment Reconciliation Script** that includes the vendor specific logic invoked by the reconciliation algorithm. The script should reference the business object **C1-StandardOneTimePaySSTask** as its schema.

  Note: Integration with Official Payments. The product provides a service script **C1-PymtRecon** that provides the additional payment reconciliation logic for integration with Official Payments.

Additional Topics

The following sections provide some additional information related to online payment functionality.

Implementing New Payment Destinations

Your implementation may wish to support additional payment options for your taxpayers. For example, perhaps a certain tax type is assigned an external identifier on its tax role and your taxpayers are able to use that identifier to "pay a given tax type". For the purposes of this section, the assumption is that the external identifier is unique across all tax roles in the system. The following information outlines the steps required in this product to support an additional payment destination.

> **Note:** Additional steps are required in the self service product. Refer to the self service product documentation for more information.

Lookup

Define a new value in the **C1_PAYMENT_TARGET_FLG** lookup to represent the new payment destination.

Create Payment Algorithm

Each payment destination requires a create payment algorithm that understands how to direct the payment to the appropriate obligation(s). In our example of paying a filing period via external id, the create payment algorithm must use a provided External ID to determine an appropriate Tax Role. From that tax role, the algorithm must determine the obligation(s) for the tax role that have outstanding debt and direct the payment towards the obligations appropriately.

The algorithm must be coded and implemented using an Algorithm Type and related Algorithm. The base product Create Payment algorithm types may be used as examples.

Characteristic Type

If there is not already a base characteristic type for each distribution detail value that must be provided for the payment distribution to work, one must be created. In this example, an Ad-hoc characteristic type to capture the External ID must be defined.

Distribution Rule

An appropriate distribution rule for each distribution detail required by the new payment destination must be created. In our example a distribution rule for "pay a given tax type" is required configuring the above created characteristic type and create payment algorithm.

> **Note:** If the payment destination requires multiple pieces of information, such as an ID plus a filing period, then multiple distribution rules are required, each referring the appropriate characteristic type. Only the last distribution rule should reference the Create Payment algorithm.

Payment Processor Service

The processor service is used for the following purposes:

- Validation. When the payment service task is first created, the processor is called to validate the details. The processor should use the payment destination details to verify that an appropriate record is found and that the appropriate account for that record can be determined. For our example, the processor should take the input External ID and find one and only one tax role with that ID and determine the tax role's account. The processor should provide appropriate errors if the data cannot be found.
- Validate Only. This is a variation of validation and is used for checking payment details when using an external vendor. The only difference between this and the validation logic is that the account ids are not retrieved for this logic.
- Build. When getting ready to process the payment, the service task BO first uses this processor to build the appropriate collection of Payment Distribution Details expected by the creation of a payment. The distribution details built by the processor are captured in the service task in the <paymentDistributionList> collection. A subsequent algorithm uses this information to create the payment with this distribution detail.

When creating a new custom Processor, use any of the base business services provided as an example of the type of functionality needed. All base business services provided begin with **C1-PaymentDestination%**. The processor may also be implemented as a service script, if preferred.

The custom service, whether it be a service script or a business service should include the data area **C1-PayDestProcessorCommon** along with a destination details group that includes the list of payment details provided by the taxpayer. These elements may be defined with specific identifiers. The service task type will contain mapping to populate the elements from the input information.

Service Task Type

Each additional payment destination requires its own service task type. Refer to *Configuration Tasks for Online Payments* for more information.

Account Verification Service

The processor is used to verify if the input account is valid for the entity identified by the current access information. If the account is invalid it returns an error. When creating a new custom Processor, use any of the base business services or service scripts provided as an example of the type of functionality needed. All base components provided begin with C1-OTPVAc%. The processor may be implemented as a service script or a business service. The custom service, whether it be a service script or a business service should include the data area C1-SelfServiceKeys along with an <accountId> element.

Retrieve Payment Due Service

When creating a new custom Processor, use any of the base business services or service scripts provided as an example of the type of functionality needed. All base components provided begin with C1-RetPymtsDue%. The processor may also be implemented as a service script, if preferred.

The custom service, whether it be a service script or a business service should include the data areas C1-SelfServiceKeys and C1-RetrievePaymentsDueCommon.

Service Task Type

Each additional payment destination requires its own service task type. See *Configuration Tasks for Online Payments* for more information.

Implementing a Fee Requirement Script

When integrating with an external payment vendor, the product provides the ability to indicate for each service task type (i.e., payment destination) if the fee should be passed onto the taxpayer or not based on taxpayer type. Your implementation may conditionally pass on the fee to certain kinds of taxpayers. For example, maybe when a taxpayer is paying a filing period, the fee is passed on if the taxpayer is not paying the filing period on time. The product provides a service script (**C1-FeeReqFPr**) that includes this logic.

If your implementation has rules for when a fee should be passed onto the taxpayer based on specific conditions, a service script with the appropriate rules must be implemented. The above base script should be used as a sample, including the input and output that is expected.

Account Information

The self service product user is viewing the tax account data and eventually pays an outstanding balance. Account information portal displays account summary, various alerts, and also filing and payment history.

This functionality follows an inquiry request handling pattern where the service task type captures various configurations and instruction but the request does not result in the service task creation. Otherwise the integration is using the inquiry request flow.

Note: Refer to the *Message Processing Overview* for more information about integration flow patterns.

The topics in this section describe more information about the provided functionality, followed by configuration tasks.

Account Summary

The tax account summary contains the most important facts your implementation may wish to bring to taxpayer's attention. The summary is supposed to provide at a glance overview of the account status.

The structure of the summary is a follows:

- **Title** - A short description of the tax account.
- **Details** - Essentials about tax account status, for example a start and end date, the date and the amount of the last payment etc.
- **Location** - An address associated with the account.
- **Current Balance** - The outstanding balance including penalty and interest forecasted as of the current date.

In the self service product the title and the details of the summary are composed using message text with the substitution parameters.

The product is expected to supply the actual data items (dates, numbers, types); this information is used by the self service product as substitution parameters for the summary messages.

The self service product sends an inquiry request containing the tax account identifiers (access type + access keys) and a line of business. The web service is processed by the XAI Inbound Services TSGetTaxAccountSummary, which invokes the service script **C1-GetAcctSm**.

The service script reads the Master Configuration, determines the service task type holding account information-related configurations and invokes an account summary retriever logic associated with request's access type.

Account Summary Retriever

The account summary retriever is expected to deliver the following information:

- Summary Title parameters list.
- Summary Details parameters list.
- Taxpayer Name (required, used internally by the self service product).
- Tax Type (optional, used internally by the self service product).
- Current balance amount and currency (optional, used internally by the self service product).
- Address data.

Retriever logic may also completely override summary title and details text and populate it on the response.

The account summary retriever provided with the base product is implemented for the access type ***Tax Role*** . It uses summary instructions defined on the service task type and returns the following details:

- Summary title parameters *tax type* and a *taxpayer name*, summary details parameters: tax role's *start date*, *outstanding balance*, last *payment's date* and *amount*.
- Account's mailing address.
- Tax role's outstanding balance forecasted as of the current date.

Note: Review the business service **C1-GetTaxRoleAccountSummary** as an example of a tax account summary retriever. Refer to *Service Task Types* for more details about summary instructions configuration.

Alerts

Alerts are meant to inform the taxpayer about various urgent matters related to the tax account and in some cases to take immediate action to resolve the issue. Alert may also be used to communicate tax legislation changes potentially affecting the taxpayer or remind about important tax-related dates.

The integration provides out of the box support for the following alerts:

- **Open collections.** This alert indicates that the tax account is associated with open collections.
- **Overdue balance exists**. This alert indicates the existence of an overdue balance for the tax account.
- **Stop Filer Alert.** This alert indicates that return is due for one of the filing periods.
- **Taxpayer Info Incomplete** This alert indicates that taxpayer record doesn't have a valid email address

The topics below describe more information about the provided functionality, followed by configuration tasks.

Retrieving Alerts

The self service product sends an inquiry request containing the tax account identifiers (access type+access keys) and a line of business. The web service is processed by the XAI Inbound Services TSGetTaxAccountAlerts, which invokes the service script C1-GetAlerts.

The service script reads the Master Configuration, determines the service task type holding alert-related configurations for the input line of business, derives applicable alert types, and invokes alert retrievers.

Alert retriever is expected to deliver the following information:

- Alert parameters list.
- Document location (optional).
- Payment Amount.

Alert parameters are used by the self service product as substitution parameters for the alert message. The document location (if provided) is displayed as a link.

Retriever logic may also completely override alert text and populate it on the response.

In the self service product many aspects of the alert are configurable, including the target navigation URL. For example, an alert may read "New child benefit claim forms are available. File now" and the link "File now" would redirect the taxpayer to the online form. The navigation URL may also be overridden by the retriever.

Note: See *Tax Account Access Information* for tax account-related inquiry flows.

Alert Type

Self service Alert Type is implemented by the product via extendable lookup. A single lookup value is representing an alert type and references alert retrievers.

The product supports two categories of alerts:

- **Access Based** alerts are alerts whose logic evaluates tax account-related issues. Alerts of this type define retrievers for each supported access type. It is implemented using the extendable lookup business object **C1-AccessTypeAlertLookup**
- **General** alerts are alerts whose logic evaluates system-wide issues. Alerts of this type define a single retriever It is implemented using the extendable lookup business object **C1-GeneralAlertLookup**

Applicable Alerts List

The tax authority has an option to designate certain alerts to a specific audience within the self service user base. The product supports the ability to configure a list of alert types applicable for a line of business, including:

- Tax account-related alerts applicable to a line of business, e.g., alerts concerning corporate taxes.
- Generic alerts applicable for a line of business, e.g., alerts concerning business registration requirements.
- Generic alerts applicable for all taxpayers.

The list is implemented using service task type business objects **C1-AccessAlertsTaskType** and **C1-GeneralAlertsTaskType**. The objects are included in the base product.

Implementing a New Alert Type

Your implementation may wish to introduce additional generic or tax account-specific alerts. The following information outlines the steps required in this product to support a new alert.

> **Note:** A new alert should be configured in the self service product. Refer to the self service product documentation for more information.

Alert Retriever

Design and implement the business logic for alert. If the reason for issuing this alert is related to outstanding payment(s), consider retrieving the amount due. Use alert parameters to communicate dates, amounts and other important information.

When creating a new custom retriever, use any of the base business services provided as an example of the type of functionality needed.

The retriever may also be implemented as a service script, if preferred.

The custom service, whether it be a service script or a business service, should include the data area **C1-TaxpayerAccountAlertCommon**.

Extendable Lookup

Define a new value in the extendable lookup to represent the new alert type.

- For tax account-related alerts, use **C1-AccessTypeAlertLookup**.
- For general alerts, use **C1-GeneralAlertLookup**.

Service Task Type

Add the new alert type to the service task type(s) that represent the applicable alerts list.

Payment History

The tax account payment history displays the list of payments. The self service product provides an ability to filter the list by a date range.

For each payment on the list the product displays payment amount, date, payment method, status and the confirmation number (where applicable).

The self service product sends an inquiry request containing the tax account identifiers (access type + access keys) and a line of business. The web service is processed by the XAI Inbound Services **TSGetPaymentHistory**, which invokes the service script **C1-GetPymtHs**.

The service script reads the Master Configuration, determines the service task type holding account information-related configurations and invokes the payment history retriever logic associated with request's access type.

The payment history retriever business service **C1-GetTaxRolePaymentHistory** provided with the base product is implemented for the access type ***Tax Role.***

Filing History

The tax account filing history displays the list of submitted forms. The self service product provides an ability to filter the list by a date range.

For each filing period the following details are retrieved:

- Filing period start and end date
- Form type and received date - not available for missing filing periods
- Filing's due date
- Filing status
- Document locator number
- Confirmation ID
- Form's printable document location
- Amount due (hidden on the self service screen)

The self service product sends an inquiry request containing the tax account identifiers (access type + access keys) and a line of business. The web service is processed by the XAI Inbound Services TSGetPaymentHistory, which invokes the service script **C1-GetPymtHs**.

The service script reads the Master Configuration, determines the service task type holding account information-related configurations and invokes the filing history retriever logic associated with request's access type.

Filing History Retriever

The filing history retriever business service **C1-GetTaxRoleFilingHistory** provided with the product is implemented for the access type ***Tax Role*** .

This retriever utilizes the following instructions defined on the service task type for account information:

- Filing types that should be excluded from the response
- Form's characteristic type for the printable form document location

- Indicates if the missing filing periods should be included in the list

The filing status is represented in the base product by the values of the lookup SS_FILING_STATUS_FLG

Configuration Tasks for Account Information

The following sections list the configuration tasks needed to implement the account information functionality.

> **Note:** For information about configuration tasks for new alert types, see *Implementing a New Alert Type*.

Characteristic Types

Printable Form Document Location

If your implementation supports the ability to link a printable document (PDF or other format) to the form and these documents are stored in the location accessible for the public, create a characteristic type of class **File Location**. Specify **Form** as a characteristic entity.

Service Task Types

Account Information Task Type

Create service task types for the **account information** configurations with the business object Account Information Task Type. Configure one service task type for each line of business your implementation wish to support.

- Select the line of business from a drop-down list.
- The related transaction BO should be set to C1-StandardInquiryTask.
- Service task class should be set to Standard Inquiry.
- To Do Type should be set to a standard error To Do type to use when transitioning to error. The base product To Do Type Self Service Task Issues (C1-SSTTD) is provided for this purpose.
- To Do Role should be configured to an appropriate default To Do role for the above To Do type. This is optional. If no value is provided the default role for the To Do type is used.
- Configure the Confirmation Header Message Category and Message Number and the Error Header Message Category and Message Number to use for communication to the self service user.

  > **Note:** Refer to the Messages section of the *General Configuration Tasks* for more information.

- **Summary Instructions.** If your implementation uses the tax account summary retrievers provided by the base product, define the instructions to use when retrieving the tax account summary:
  - Specify a default name type to be used as a taxpayer name on the summary.
  - Indicate if the address should be included in the summary by default. This configuration could be different for ***Business*** and ***Individual*** line of businesses. For example, an individual is likely to have a single mailing address associated with all tax roles and accounts and there is no need to display the address redundantly in multiple places.
  - List special instructions for the tax types supported by your implementation:
    - Use checkbox to indicate if the primary taxpayer's data should be used for the summary.

    - If the above is unchecked, specify either account-person relationship type or tax role-person relationship type to use in order to find a taxpayer whose information should be used.
    - Specify the taxpayer's name type that should be displayed.
    - Indicate if the summary should include the address. For example, consider the situation where Individual line of business is encompassing both individual income and property taxes. For the individual income, the address bears little significance but for the real property tax it makes sense to display the address associated with the tax roles.
- **Filing History Instructions.** If your implementation uses the tax account filing history retrievers provided by the base product, define the instructions to use when retrieving the tax account fling history:
  - Specify a char type created above as a Char Type for Printable Form Location.
  - Indicate if missing filing periods should be included in the list.
  - List filing types to exclude from the filing history.
- **Payment History Instructions.** If your implementation uses the tax account payment history retrievers provided by the base product, define the instructions to use when retrieving the tax account payment history:
  - Indicate if cancelled payments should be included in the list.
- **Access Type Processors.** For each supported access type list the processors responsible for the tax account information retrieval.
  - Account Summary Retriever.
  - Filing History Retriever.
  - Payment History Retriever.

    Note: The base product provides the business services **C1-GetTaxRoleAccountSummary**, **C1-GetTaxRolePaymentHistory**, and **C1-GetTaxRoleFilingHistory**. See the *Account Summary*, *Filing History*, and *Payment History* topics for more details.

Applicable Alerts Task Type

Create service task types for the **general** and **tax account-related alert lists** with the business objects **General Alerts** and **Access Based Alerts**, respectively. Use this configuration to group the alerts according to your business requirements.

- Select the line of business from the drop-down list.
- The related transaction BO should be set to C1-StandardInquiryTask.
- Service task class should be set to Standard Inquiry.
- To Do Type should be set to a standard error To Do type to use when transitioning to error. The base product To Do Type Self Service Task Issues (C1-SSTTD) is provided for this purpose.
- To Do Role should be configured to an appropriate default To Do role for the above To Do type. This is optional. If no value is provided the default role for the To Do type is used.
- Configure the Confirmation Header Message Category and Message Number and the Error Header Message Category and Message Number to use for communication to the self service user.

  Note: See the Messages section of the *General Configuration Tasks* topic for more information.

- **Alert Types**. List all alert types applicable for the line of business selected above.

Master Configuration

The self service master configuration record includes several settings required for the account information inquiry processing:

- Account Information. For each supported line of business (as defined in the LINE_OF_BUSINESS_FLG lookup) indicate the corresponding Service Task Type for Account Information.
- Alerts.
  - Specify a Service Task Type that defines default generic alert types list. If no line of business is included in the alert inquiry web service request, this list will be used.
  - For each supported line of business (as defined in the LINE_OF_BUSINESS_FLG lookup) indicate the corresponding Service Task Types for tax account-related and generic alerts.

Integration Mapping

Alert Types. All supported alert types must be configured in the alert type domain value mapping in the SOA Composer application.

Tender Types. Payment captured in the system and shown on the payment history may be coming from various sources, not limited to the self service product. In order to display all payments properly all Tender Types defined in the system must be configured in the tender type domain value mapping in the SOA Composer application.

Payment Status. Payment statuses used in the system must be configured in the payment status domain value mapping in the SOA Composer application.

Refer to the self service product documentation for details.

Taxpayer Information

The self service product user is viewing taxpayer information and allowed to make certain updates online. The taxpayer in this case is a taxpayer linked to the tax account in context, which means that "user" is not always equal to a "taxpayer"; for example consider the situation where the business owner is viewing the business account and exploring related business taxpayer record as opposed to the same person viewing his/her own personal income tax account and related individual's information.

Taxpayer portal includes:

- The *taxpayer summary*
- The *contact information*
- The *correspondence information*

This functionality uses the service task types to captures various configurations and instruction but using it may also result in the service task creation.

Note: See *Message Processing Overview* for more information about integration flow patterns.

The topics below describe more information about the provided functionality, followed by configuration tasks.

Taxpayer Summary

The taxpayer summary contains the most important facts your implementation may wish to bring to user's attention. The structure of the summary is a follows:

- **Title** - a short description of the taxpayer
- **Details** - essential details.
- **Primary Contact -** the name and an email address of a primary contact

In the self service product the title and the details of the summary are composed using message text with the substitution parameters.

The base product is expected to supply the actual data items - dates, numbers, types; this information is used by the self service product as substitution parameters for the summary messages.

The self service product sends an inquiry request containing the tax account identifiers (access type + access keys) and a line of business. The web service is processed by the XAI Inbound Services **TSGetTaxpayerSummary**, which invokes the service script **C1-GetTxpSum**.

The service script reads the Master Configuration, determines the service task type holding taxpayer information-related configurations and invokes the taxpayer summary retriever logic associated with request's access type.

Taxpayer Summary Retriever

The account summary retriever is expected to deliver the following information:

- Summary Title parameters list
- Summary Details parameters list
- Taxpayer Name (required, used internally by the self service product)
- Taxpayer Type (optional, used internally by the self service product)
- Primary Contact details:
  - Contact Type
  - Contact Name
  - Email address

Retriever logic may also completely override summary title and details text and populate it on the response.

The taxpayer summary retriever provided with the base product is implemented for the access type ***Tax Role*** . It uses summary instructions defined on the service task type to return the following details:

- Summary title parameters *taxpayer name* and *ID number*, summary details parameters: *taxpayer type*
- Primary Contact details, where applicable.

Review the business service **C1-RetrieveTaxpayerInfoSummary** for an example of a taxpayer summary retriever. Refer to *Service Task Types* for more details about summary instructions configuration.

Contact Information

Contact information includes a list of taxpayer's phone numbers and an email address stored in the system. The self service user may choose to add and/or edit the information.

The self service product sends a request containing the tax account identifiers (access type + access keys) and a line of business. It may also contain updated contact info. The web service is processed by the XAI Inbound Services **TSGetTaxpayerContactInformation**, which invokes the service script **C1-GetTxpCon**.

The service script reads the Master Configuration and determines the service task type holding taxpayer information-related configurations.

The contact info instruction on the service task type indicates whether it should be the taxpayer itself or the person linked to the taxpayer via person-to-person relationships.

The script proceeds according to the input action:

- For action **READ** it invokes the contact info retriever logic associated with request's access type
- For action **UPDATE** it creates the service task referenced on the service task type. Service task algorithm(s) perform the actual update of the taxpayer information.

Note: Refer to the algorithm **C1-UPDTXPINF** for more details about taxpayer contact info update logic.

Contact Info Retriever

The taxpayer contact info retriever provided with the base product is implemented for the access type ***Tax Role*** . It uses contact info source instructions defined on the service task type to return the following details:

- List of taxpayer's phone numbers, including extensions.
- Email address.

The instruction indicates whether the source of the contact info is the input taxpayer itself or the person linked to the taxpayer via person-to-person relationships.

Review the business service **C1-RetrieveTaxpayerContactInfo** for an example of a taxpayer contact info retriever. See *Service Task Types* for more details about contact information source instructions configuration.

Correspondence Information

The correspondence information contains list of addresses associated with the taxpayer. The self service user may choose to add and/or edit the specific address.

The portal displays address type and formatted address info for each address instance.

The base product is expected to supply the actual address fields and the formatting is performed in the self service product.

The self service product sends a request containing the tax account identifiers (access type + access keys) and a line of business. The web service is processed by the XAI Inbound Services **TSGetTaxpayerCorrespondenceInformation**, which invokes the service script **C1-GetTxpCor**.

The service script reads the Master Configuration, determines the service task type holding taxpayer correspondence-related configurations and invokes the address retriever logic associated with request's access type

The correspondence info retriever provided with the base product is implemented for the access type ***Tax Role*** .

Review the business service **C1-GetTaxpayerCorrespondInfo** for an example of a taxpayer address information retriever. Refer to *Service Task Types* for more details about related configurations.

Update an Address

The self service product allows taxpayer to update an existing address record.

The integration applies the following steps:

- The self service user modifies existing address information or adds a new address.
- The self service product sends a request containing the tax account identifiers (access type + access keys), line of business, and the address data. The web service is processed by the inbound web service **TSAddressMaintenance**, which invokes the service script **C1-MaintAddr**.
- The service script reads the Master Configuration and determines the service task type holding taxpayer correspondence-related configurations for the input line of business.
- It then creates the service task referenced on the service task type. Service task algorithm(s) perform the actual update of the address.

  Note: Refer to the algorithm **C1-MNTTXPADR** for more details about taxpayer correspondence information update logic.

Configuration Tasks for Taxpayer Information

The following sections list the configuration tasks needed to implement the taxpayer information functionality.

Service Task Types

Taxpayer Information Task Type

Create service task types for the **taxpayer information** configurations with the business object Taxpayer Information Task Type. Configure one service task type for each line of business your implementation wishes to support, as follows:.

- Select the line of business from the drop-down list.
- The related transaction BO should be set to **C1-UpdateTaxpayerInfoTask**.
- The service task class should be set to **Service Request**.
- **To Do Type** should be set to a standard error To Do type to use when transitioning to error. The base product provides **To Do Type Self Service Task Issues** (C1-SSTTD) for this purpose.
- **To Do Role** should be configured to an appropriate default To Do role for the above To Do type. This is optional. If no value is provided, the default role for the To Do type is used.
- Configure the **Confirmation Header Message Category and Message Number** and the **Error Header Message Category and Message Number** to set up communication with the self service user.

  Note: See the Messages section of the *General Configuration Tasks* for more information.

- **Summary Instructions.** If your implementation uses the taxpayer summary retrievers provided by the base product, define the instructions to use when retrieving the taxpayer summary:

  - Specify a default name type to be used as a taxpayer name on the summary
  - List special instructions for the taxpayer types supported by your implementation. This configuration could be different for ***Business*** and ***Individual*** line of businesses. For example, individual taxpayer may not need/have an additional person to be listed as primary contact, while businesses are likely to have a designated person for this purpose:
    - Use the checkbox to indicate if the primary taxpayer's data should be used as the primary contact.
    - If the above is unchecked, specify either person-to-person relationship type to use in order to find a taxpayer whose information should be used.
    - Specify taxpayer's name type that should be displayed for the primary contact.
- **Contact Info Source Instructions.** If your implementation uses the taxpayer contact info retrievers provided by the base product and the algorithms linked to the base business object, define the instructions below. Note that the taxpayer whose contact information is displayed and updated is not necessarily the same taxpayer whose email was used for the primary contact. Consider the following situation: the taxpayer "in context" is a business. The source of the primary contact for the business could be a customer service department (person) linked to the main business (person). However the source of the contact info should be a business (person) itself, with phone numbers of business's departments.
  - List special instructions for the taxpayer types supported by your implementation:
    - Use the checkbox to indicate if the primary taxpayer's data should be displayed and updated online.
    - If the above is unchecked, specify either person-to-person relationship type to use in order to find a taxpayer whose information should be used
- **Access Type Processors.** For each supported access type list the processors responsible for the tax account information retrieval.
  - Taxpayer Summary Retriever.
  - Contact Info Retriever.

> **Note:** The product provides out of the box the business services **C1-RetrieveTaxpayerInfoSummary** and **C1-RetrieveTaxpayerContactInfo**.

Taxpayer Correspondence Information Task Type

Create service task types for the **correspondence information** configurations with the business object Taxpayer Addresses Task Type. Configure one service task type for each line of business your implementation wish to support.

- Select the line of business from the drop-down list.
- The related transaction BO should be set to C1-GetTaxpayerCorrespondInfo.
- Service task class should be set to Service Request.
- To Do Type should be set to a standard error To Do type to use when transitioning to error. The base product To Do Type Self Service Task Issues (C1-SSTTD) is provided for this purpose.
- To Do Role should be configured to an appropriate default To Do role for the above To Do type. This is optional. If no value is provided the default role for the To Do type is used.
- Configure the Confirmation Header Message Category and Message Number and the Error Header Message Category and Message Number to use for communication to the self service user.

> **Note:** See the Messages section of the *General Configuration Tasks* for more information.

- **Access Type Processors.** For each supported access type list the processors responsible for the tax account information retrieval.

  Correspondence Info Retriever

Note: The base product provides the business services **C1-GetTaxpayerCorrespondInfo**.

Master Configuration

The self service master configuration record includes settings required for the account information inquiry processing:

- **Taxpayer Information**. For each supported line of business (as defined in the LINE_OF_BUSINESS_FLG lookup) indicate the corresponding Service Task Type for Taxpayer Information.
- **Correspondence Information**. For each supported line of business (as defined in the LINE_OF_BUSINESS_FLG lookup) indicate the corresponding Service Task Type for Correspondence Information.

Integration Mapping

Phone Types. All Phone Types defined in the system must be configured in the phone type domain value mapping in the SOA Composer application.

Alternatively, your implementation may choose to use exactly same values in both products.

Online Form Processing

In this release, the integration supports on-line form filing. The following diagram illustrates an expected process flow:

The integration has multiple steps.

Initial processing works as follows:

- **Casual user.** The self service application captures information about the taxpayer and sends a web service request to this product to validate the taxpayer (proof of identity - POI).
- **Enrolled user.** The identification is not performed.

Note: See *User Identity Verification* for more information.

The interpretation of the identification outcome includes a provision for the first-time filers. The option is configured in the self service product. If first-time filers are allowed, the identification request may return neither taxpayer ID(s) nor an identification error.

Next the online form is rendered and the taxpayer begins to fill in the information. User may press action buttons to request the data to be copied from the previous return and validate the incomplete form multiple times. Every action is triggering a web service call and the base product is processing the incoming web service request. See *Form Actions* for more details.

Lastly, the user is pressing Ready To File button and the form data is validated one last time. If this action produced no errors, the user is offered an option to review and print the form and then submit it. The integration generates the DLN and also a confirmation ID.

As a result of a successful online form submission the form is created and linked to the form submission service task. The DLN is captured on the service task via characteristic.

The topics below describe more information about the provided functionality, followed by configuration tasks.

Form Data Model

Form definitions are in a sense shared between self service application and the base product. Both systems rely on the metadata when rendering forms on screen and applying various business rules on the form data.

The solution provides the ability to import form type definitions into the self service product.

The self service product operates with generic form data model. The requests originated from the self service product contain the form data in this format:

```
<formData>
<id/>
<name/>
<section type="list">
<id/>
<name/>
<sequence/>
<line type="list">
<field type="group">
<id/>
<name/>
<dataType/>
<value/>
</field>
</line>
<table type="list">
<id/>
<name/>
<tableRow type="list">
<sequence/>
<field type="list">
<id/>
<name/>
<dataType/>
<value/>
</field>
</tableRow>
</table>
</section>
</formData>
```

The form processing integration flow assumes that for the form of a certain type the names of the elements on the generic form data model (sections, lines, tables etc) exactly match the business names of the elements on the form type definitions. This assumption allows an automated transformation of the generic form data XML into form business object XML based on the form type definition.

The base product provided with the business services **C1-TransformFormToGenericWeb** and **C1-TransformGenericWebToForm** for the form data conversion.

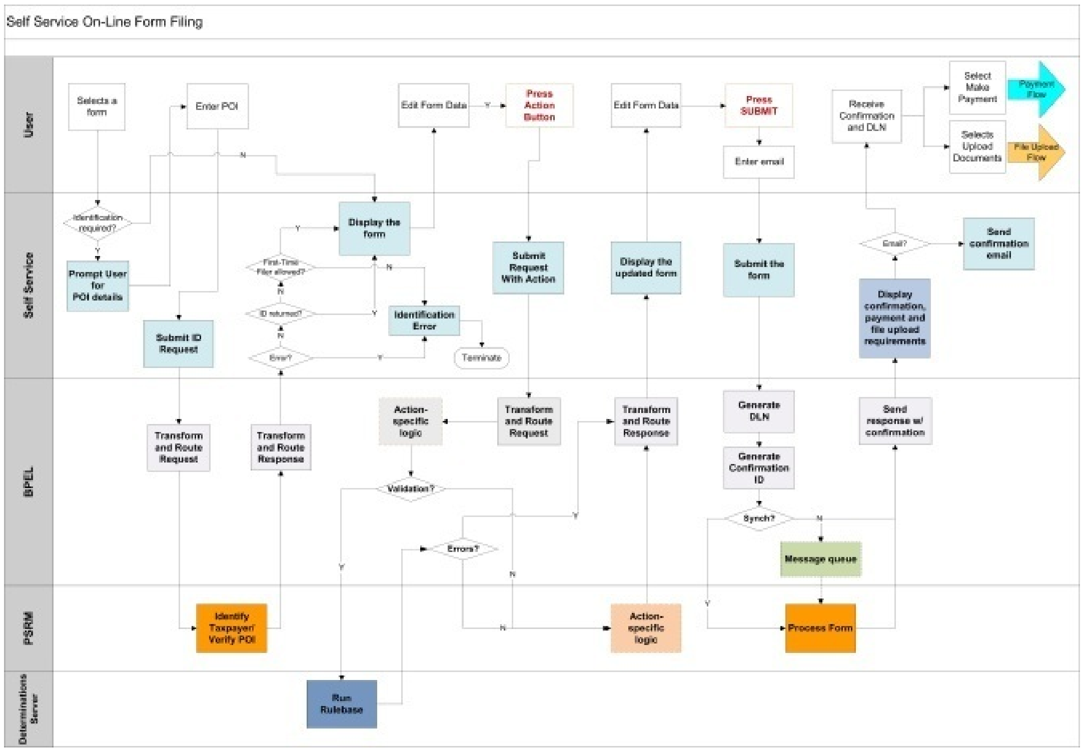
Fastpath: Refer to the *Import Form Definition* for more information

Form Process Request

The web services handling online business registration and tax form filing are processed by the XAI Inbound Services **TSProcessRegistrationForm** and **TSProcessTaxForm**, respectively. Both services invoke the service script **C1-RSISvcReq**. The appropriate service task type is derived from the Master Configuration based on the information received from the self service system.

The handling of a specific form action is delegated to the processor components (service scripts or business services) specified on the Master Configuration.

The following diagram illustrates the online form filing flow:



Form Actions

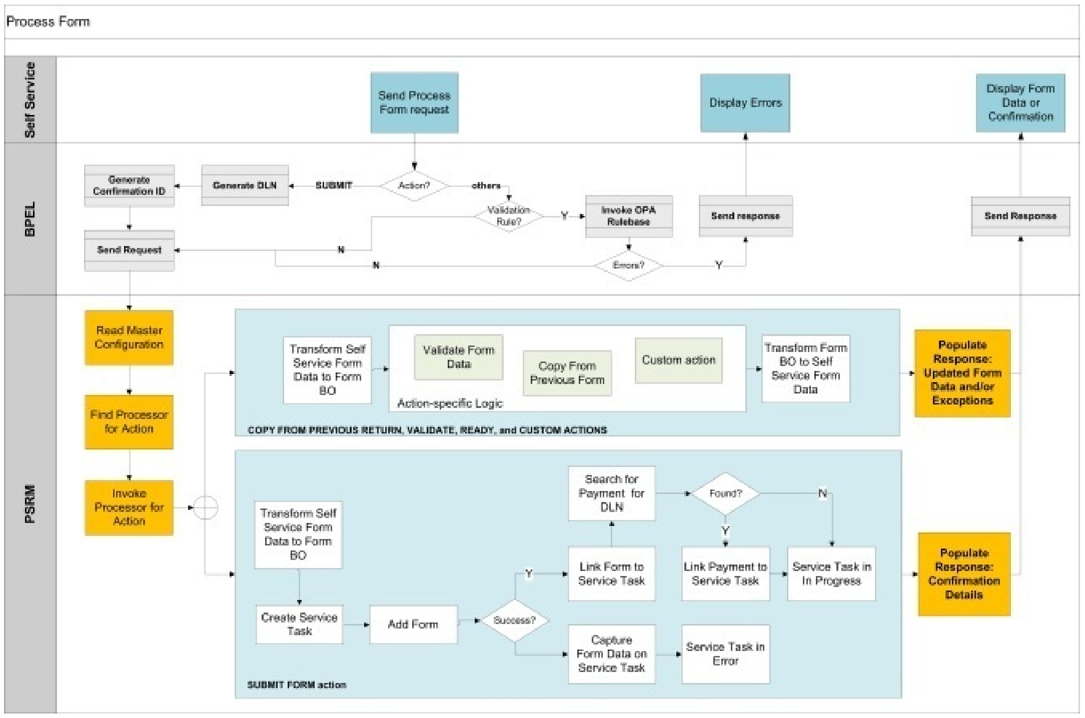

Process Form
Self Service
BPEL
PSRM
Send Process Form request
Display Errors
Display Form Data or Confirmation
Generate Confirmation ID
Generate DLN
SUBMIT
Action?
others
Validation Rule?
Y
Invoke OPA Rulebase
Send response
Send Response
Send Request
N
N
Errors?
Y
Read Master Configuration
Find Processor for Action
Invoke Processor for Action
Transform Self Service Form Data to Form BO
Validate Form Data
Copy From Previous Form
Custom action
Action-specific Logic
Transform Form BO to Self Service Form Data
Populate Response: Updated Form Data and/or Exceptions
COPY FROM PREVIOUS RETURN, VALIDATE, READY, and CUSTOM ACTIONS
Search for Payment for DLN
Found?
N
Y
Transform Self Service Form Data to Form BO
Link Form to Service Task
Link Payment to Service Task
Service Task in In Progress
Populate Response: Confirmation Details
Y
Create Service Task
Add Form
Success?
Capture Form Data on Service Task
Service Task in Error
SUBMIT FORM action

Figure 2: Handling Form Actions in PSRM

The following form actions are supported by the base product.

Copy From Previous Return is expected to return an updated form data or a message indicating that no previous return was found. The product supports an alternative optional scenario for this action. If no previous return is found, the form may be populated with basic taxpayer details, e.g., primary ID, name, and mailing address. This is done by implementing the action **Get Taxpayer Info**.

Note: The action **Get Taxpayer Info** is *not* called directly from the self service application.

Check Form Data. This action triggers the form data validation and the response is expected to contain the corrected form data and the list of exceptions. The solution provided in the base product performs the validation using the **C1-ApplyWSSValidation** business service.

Ready to File. This action is expected to perform the final round of the form data validations. The outcome is expected to be similar: updated form data and a list of exceptions, if any. If this action returns no errors/exceptions, the self service product allows the user to submit the form.

Create Form. This action is responsible for actual form creation. The service task created as a result of a form submission plays a slightly different role than other service tasks. The form is created "outside" of the task; the form action processor **C1-CreateForm** creates the form and links it to the service task as a related object.

If the form submission is followed by supporting documents, upload, and/or payments, these entities are also linked to the service task. Thus, the service task serves as the tracking device for the various activities related the form submission. The lifecycle of the service task business object includes a monitor algorithm that completes the service task when the form is successfully posted.

Note: All form action processing starts with transforming the self service generic form data XML into base form business object XML and ends with transforming the form business object XML into a self service generic form data XML. The business logic is performed on the form business object.

Notes on Form Validation

The business service **C1-ApplyWSSValidation** invokes the form rules linked to the WSS Validation rule event. When designing the rules, consider the following circumstances:

- Calculate whatever is possible to calculate and minimize manual input.
- Auto-correct is the preferable rule outcome. Create an exception only if the self service user could possibly fix the issue.
- The form may be only partially filled when the user invokes the validation.
- The validation is likely to be invoked multiple times.
- Don't include internal IDs and other sensitive information in exception messages.
- By default, the self service product displays the expanded message text for each rule exception. Make sure these messages are suitable for the self service user; explain the issue in a user-friendly manner and avoid technical details.

Configuration Tasks for Online Form Processing

The following sections list the configuration tasks needed to implement the on-line form processing functionality.

Form Sub-Type

The form sub types defined in the base product - ***Business Registration Form*** and ***Tax Form*** - must be configured in form category domain value mapping in the SOA Composer application. Refer to the self service product documentation for details.

Form Rule Group

The self service user is certainly less competent than the agency user and would likely make various mistakes and miscalculate the numbers. Even though the on-line form is equipped with multiple help and guidance tools and the form data is validated using Oracle Policy Automation-based validation engine, the chances are that the self service user would repeat the *Check My Form* action multiple times. When designing the form rule group(s) associated with WSS Validate (C1WI) rule event try to maximize the auto-correction and to minimize the number of possible exceptions.

Form Source

You may wish to create a dedicated form source for the forms submitted through the self service portal

Service Task Types

Taxpayer Identification Task Type

Define a task type to use for taxpayer identification with the business object Taxpayer Service Request Self Service Task Type.

- The **Related Transaction** BO should be set to **C1-TaxpyrIdentFormFilingSSTask**.

  Note: The state algorithm **C1-IDTXFORMF** implements the first-time filer scenario.

- **Service Task Class** should be set to **Taxpayer Identification**.
- **To Do Type** should be set to a standard error To Do type to use when transitioning to error. The base product To Do Type Self Service Task Issues (**C1-SSTTD**) is provided for this purpose. Note that the algorithm plugged in on the error state for the base business object checks that there is not already a To Do for the record before creating one of this type. This allows algorithms that detect an error to also create a To Do with a specific type, if desired.
- **To Do Role** should be configured to an appropriate default To Do role for the above To Do type. This is optional. If no value is provided, the default role for the To Do type is used.
- Configure the Confirmation Header Message Category and Message Number and the Error Header Message Category and Message Number to use for communication to the self service user.

  Note: Refer to the **Messages** section of the General Configuration Tasks for more information.

- The service request fields mapping is used when creating the target service task to correctly populate the **requestField** list from the list of fields passed in on the web service request XML. For the base transaction business object, the following fields are expected:

Sequence	Transaction BO XPath
1	requestData/legalName
2	requestData/idType
3	requestData/personIdNumber

- Configure the Service Request Data Instructions. Specify if the identification request task should be persistent and whether the web service response should be echoing back the input data.

  Note: Refer to the Messages section of the General Configuration Tasks for more information.

Note: Service Task Type Mapping. The service task type defined here must be configured in the service task domain value mapping in the SOA Composer application. Refer to the self service product documentation for details.

Form Submission Task Types

Define a task type to use for each Form Sub Type (***business registration*** and ***tax***) with the business object **Form Submission Self Service Task Type**.

- The related transaction BO should be set to **C1-RegFormSubmissionSSTask** and **C1-TaxFormSubmissionSSTask**, respectively.
- The service task class should be set to *Standard Self Service*.

- **To Do Type** should be set to a standard error To Do type to use when transitioning to error. The base product provides **To Do Type Self Service Task Issues** (C1-SSTTD) for this purpose.
- **To Do Role** should be configured to an appropriate default To Do role for the above To Do type. This is optional. If no value is provided, the default role for the To Do type is used.
- Configure the **Confirmation Header Message Category and Message Number**, as well as the **Error Header Message Category and Message Number**, to enable communication to the self service user.

  Note: See the *Messages* section of the *General Configuration Tasks* topic for more information.

Master Configuration

The self service master configuration record includes several settings required for the form definition import processing.

- Set Char Type for DLN to **C1-DLNSS**.
- Specify the Form Source created above.
- Configure the service task types to use when processing for each form sub-type. Create entries for Business Registration Form and Tax Form sub-types and specify service task types created above.
- Configure the processing components for all form actions supported by the base product. The business services mentioned below are provided with the product. The implementation may use them as-is or implement alternative action processors.
  - Action **Copy From Previous Return** - specify a business service **C1-CopyFromPreviousForm**.
  - Action **Ready to File** - specify a business service **C1-ValidateFormData**.
  - Action **Create Form** - specify a business service **C1-CreateForm**.
  - Action **Check Form Data** - specify a business service **C1-ValidateFormData**.
  - Action **Get Taxpayer Information** – specify a business service **C1-GetTaxpayerInfoForm** if your implementation is to support the special behavior for the **Copy From Previous Return** action, where basic taxpayer information is populated on the form if no previous return is found.

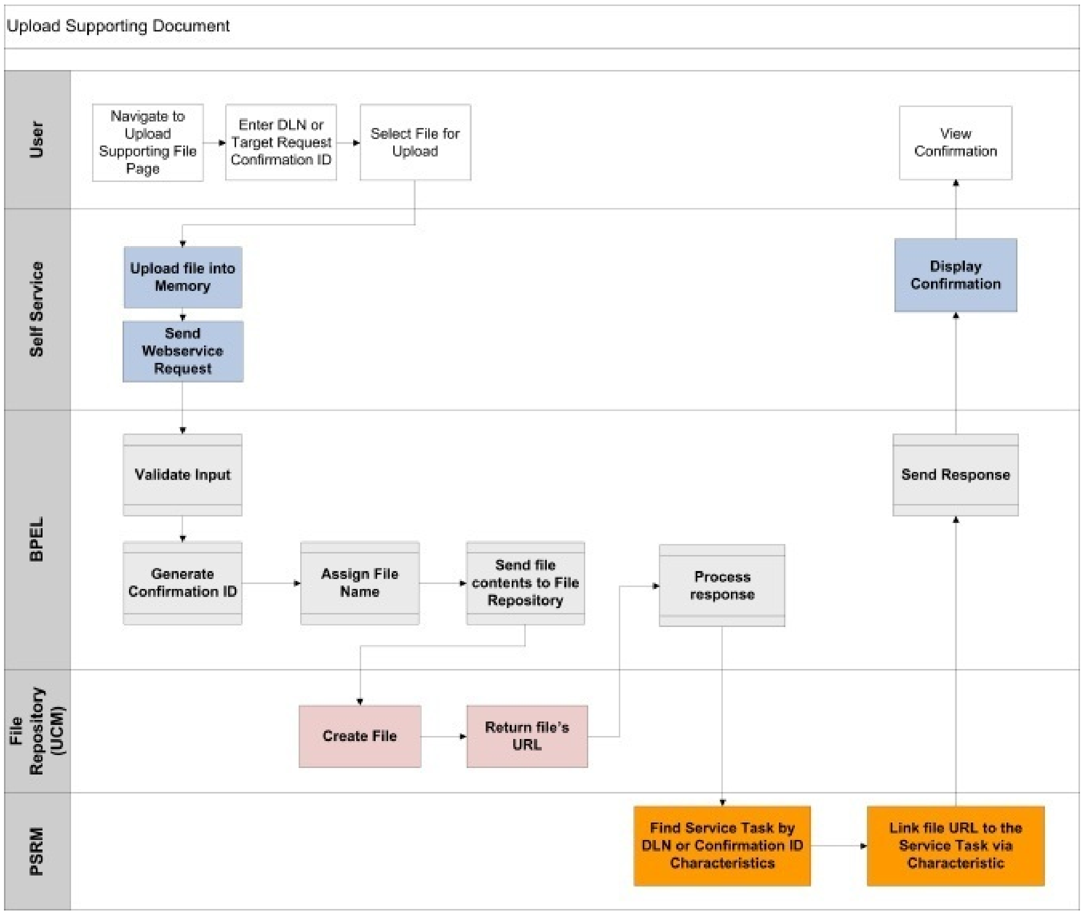
Implementing a Custom Form Action

Your implementation may wish to support additional actions for the online filing process. The self service product allows enabling a form action on specific form. For example, consider the following requirement/implementation scenario:

- **Requirement:** Improve the self service user's experience by splitting **Check My Form** into two actions: *Calculate Totals*, whose purpose is to update the numbers on the form, and *Check my Form* whose purpose is purely data verification.
- **Possible implementation:**
  - Create a new **Form Rule Event** named *Calculate* and implement form rules for calculations.
  - Implement the form action processor to call a rule event created above **C1-ApplyWSSValidation**. For the base product, from XML conversion to and from the self service generic XML, use the business services **C1-TransformFormToGenericWeb** and **C1-TransformGenericWebToForm**.
  - Add a new form action using lookup **C1_SELF_SVC_FORM_ACTION_FLG**.
  - Configure the new action on the **Self Service Master Configuration**.
  - Set up a new action in the self service product and enable it on the form definition(s).

Upload Supporting Documents

The following diagram illustrates the Upload Supporting Documents process flow:



The integration supports an ability to upload supporting documentation for a form or a service request with the following steps:

- The file is uploaded on the self service portal and processed by the SOA Composite.
- File contents are sent to the document repository and the result file location is populated on the request.
- A confirmation ID for the file upload event is generated and populated on the request. The request is forwarded to the base product.

The request is processed by the XAI Inbound Services **TSUploadSupportingDocument**, which invokes the service script **C1-UplSupFle** .

The service script is searching for the service task either by a target confirmation ID or a DLN. The logic is adding a characteristic to the service task referencing the input supporting file URL and adds a new detail to the service task confirmation info containing the file upload confirmation ID.

Configuration Tasks for Supporting Documents Upload

The following sections list the configuration tasks needed to implement the upload supporting files for form functionality.

Master Configuration

The self service master configuration record includes settings required for the form definition import processing.

Specify characteristic type C1-SUPDC as a Char Type for File Location.

Import Form Definition

The integration supports the import of form type definitions into the self service product. The ability to import the definition allows seamless integration between the product and the self service application.

The import process has the following steps:

- The implementer enters a form sub-type and other search criteria. This triggers the web service call.
- The request is processed by the XAI Inbound Service **TSRetrieveActiveFormTypes**, which invokes the service script **C1-RetAcFrTy**. The service script reads the Master Configuration and invokes the defined processor. The list of form types available for import is determined by the product and returned back to the self service portal.
- The user is prompted to select one or more form types. This triggers another web service call for the service **TSRetrieveFormTypeDefinitions**, and the form definitions are retrieved and captured as a self service product's form definition.

The imported information for a single form type includes:

- **Form Sections:** Business name, display sequence, occurrence, label.
- **Form Single Lines:** Business name, display sequence, line number, label, data type, referenced lookup (if applicable), data precision, and scale.
- **Form Group Lines:** Business name, display sequence, line number, label.
- **Referenced lookups:** A list of values and descriptions for each lookup referenced by a form line.

Notes:

- Form Group Line is interpreted as a table in self service product. On the import XML it is represented by the tag <table>.
- Form Single Line within a group line is interpreted as a table column.
- All language-base information is returned for all languages supported by the base product.

Configuration Tasks for Import Form Definition

The following sections list the configuration tasks needed to implement the form definition import functionality.

Master Configuration

The self service master configuration record includes settings required for the form definition import processing.

Specify the business service **C1-RetrieveActiveFormTypes** as Active Form Types Retriever.

Chapter 4

Maintaining Self Service Tasks

Use the Service Task transaction to view and maintain service tasks. Navigate using **Main Menu** > **Self Service** > **Service Task.**

Self Service Task Query

Use the query portal to search for a service task. Once a service task is selected, you are brought to the maintenance portal to view and maintain the selected record.

Self Service Task Portal

This page appears when viewing the detail of a specific self service task.

Self Service Task - Main

This portal appears when a Self Service Task has been selected from the Self Service Task Query portal.

The topics in this section describe the base-package zones that appear on this portal.

Self Service Task

The Self Service Task zone contains display-only information about the selected Self Service Task, including the following:

- Audit information for the task, including the web user ID and name, the email address and IP address.
- The list of related objects associated with the task
- Any error information

- Confirmation ID and message and any confirmation details
- Other information that is specific for the type of task. Refer to the in-line help text for more information about the fields.

Self Service Task - Log

Navigate to the Log tab to view the logs for a self service task.