“WEB SERVICES”

Async Web Services

url: https://agilereporting.us.dell.com:7002/AgileAsyncSWService/services/AgileAsyncSWService

wsdl: https://agilereporting.us.dell.com:7002/AgileAsyncSWService/services/AgileAsyncSWService?wsdl

Credentials:

- Username: Americas\serviceagileprod
- Password: *********************

requestDUPFileAction

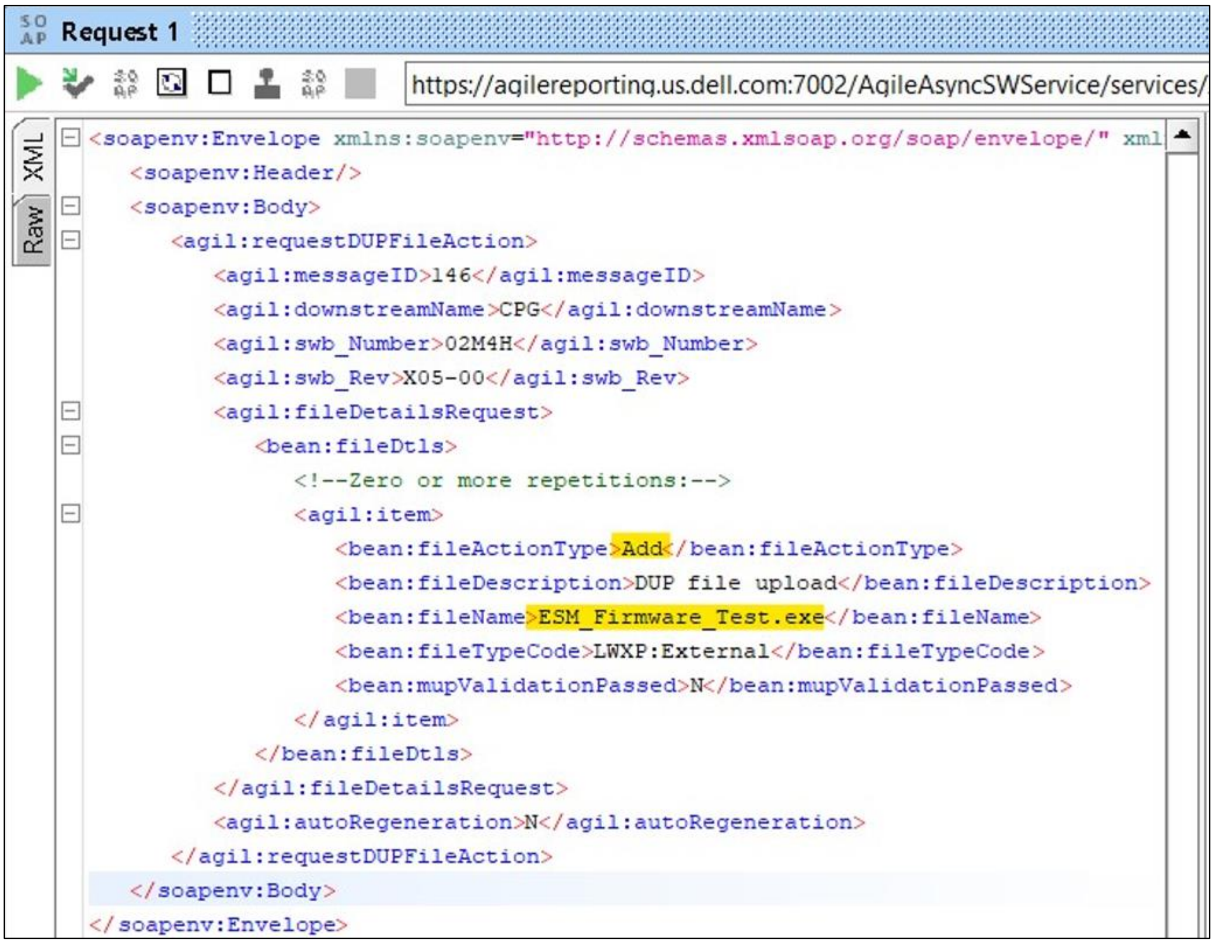
- Method Description:
  - The method will take the request from downstream for file action Add, Delete /or Replace and Auto Regeneration and return the response if the action was successful/unsuccessful.

- Working:
  - Instantiate RequestDUPFileActionValidations.
  - Perform basic validations using basicValidations method.
  - If basic validations fail, throw a ValidationException.
  - Create an Agile session using createSession method.
  - Perform file data validations using fileDataValidations method.
  - If file data validations fail, throw a ValidationException.

- Parameters:
  - String messageID,
  - String downstreamName,
  - String swb_Number,
  - String swb_Rev,
  - FileDetailsRequest fileDetailsRequest,
  - String autoRegeneration

- Response:
  - Response will content status Success/Failure along with ObjectID and Validation Error.

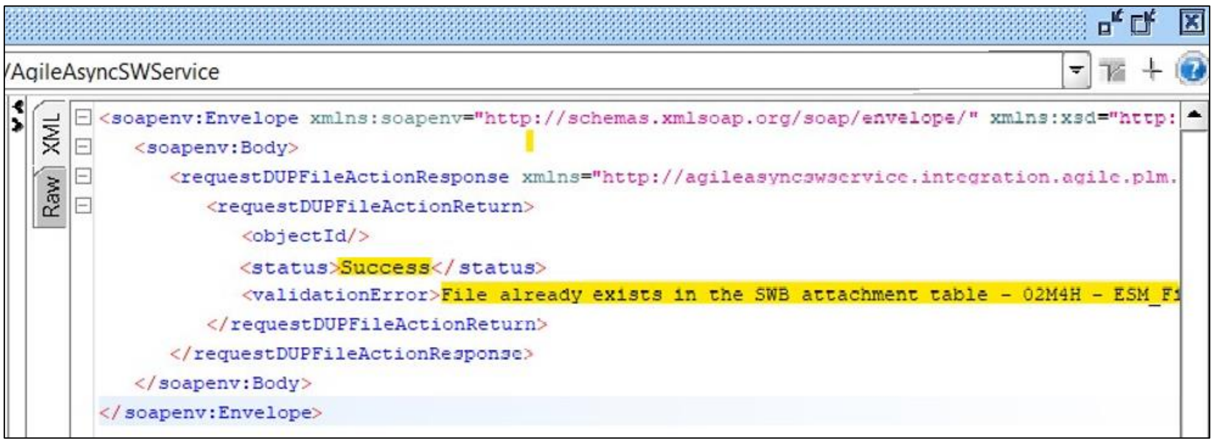
- Example:
  - Request from Downstream



```
https://agilereporting.us.dell.com:7002/AgileAsyncSWService/services/
```

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xml
   <soapenv:Header/>
   <soapenv:Body>
      <agil:requestDUPFileAction>
         <agil:messageID>146</agil:messageID>
         <agil:downstreamName>CPG</agil:downstreamName>
         <agil:swb_Number>02M4H</agil:swb_Number>
         <agil:swb_Rev>X05-00</agil:swb_Rev>
         <agil:fileDetailsRequest>
            <bean:fileDtls>
               <!--Zero or more repetitions:-->
               <agil:item>
                  <bean:fileActionType>Add</bean:fileActionType>
                  <bean:fileDescription>DUP file upload</bean:fileDescription>
                  <bean:fileName>ESM_Firmware_Test.exe</bean:fileName>
                  <bean:fileTypeCode>LWXP:External</bean:fileTypeCode>
                  <bean:mupValidationPassed>N</bean:mupValidationPassed>
               </agil:item>
            </bean:fileDtls>
         </agil:fileDetailsRequest>
         <agil:autoRegeneration>N</agil:autoRegeneration>
      </agil:requestDUPFileAction>
   </soapenv:Body>
</soapenv:Envelope>
```

  - Response from Agile



```
/AgileAsyncSWService
```

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsd="http:
   <soapenv:Body>
      <requestDUPFileActionResponse xmlns="http://agileasyncswservice.integration.agile.plm.
         <requestDUPFileActionReturn>
            <objectId/>
            <status>Success</status>
            <validationError>File already exists in the SWB attachment table - 02M4H - ESM_Fi
         </requestDUPFileActionReturn>
      </requestDUPFileActionResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

requestFileDownload

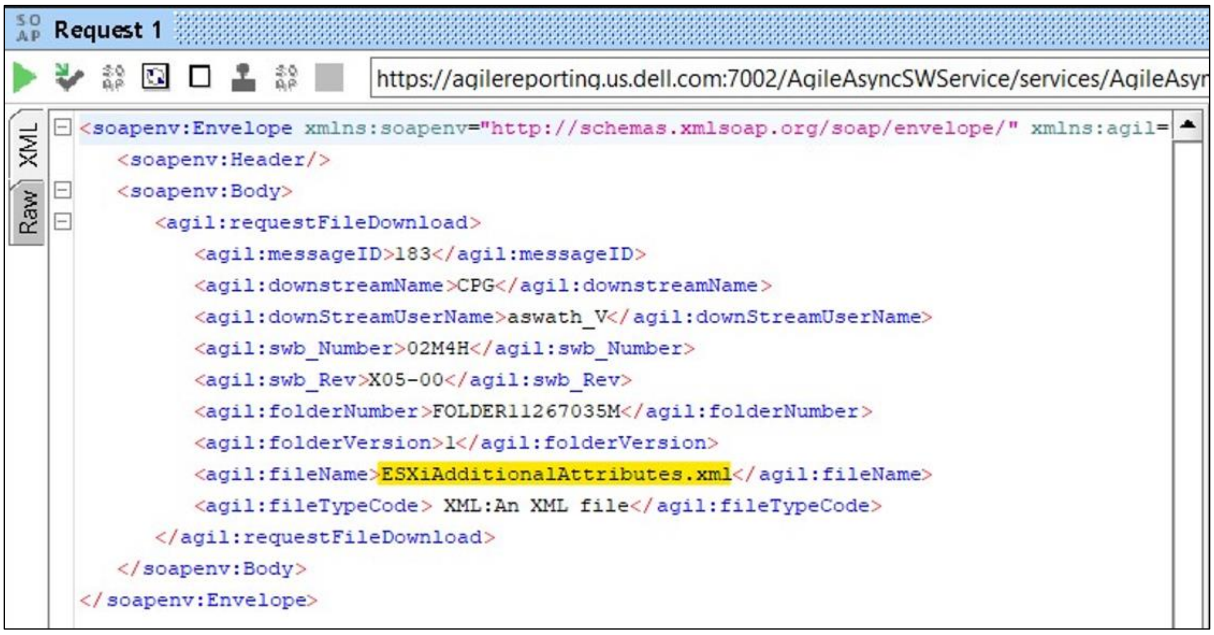
- Method Description:
    - The method is download request exposed to downstream.
    - requestFileDownload calls to method requestRCDFileDownload to perform validation.
    - AddServiceQueueDAO dao object is created and call to method addServiceQueue.
    - addServiceQueue method takes all the parameters process it to create a query and execute to enter a record into the database.

- Working:
    - The requestFileDownload method calls requestRCDFileDownload, reducing the complexity by setting default values for some of the parameters.
    - This allows for a simpler interface when those additional parameters are not necessary for the file download request.

- Parameters:
    - String messageID,
    - String downstreamName,
    - String downStreamUserName,
    - String swb_Number,
    - String swb_Rev,
    - String folderNumber,
    - String folderVersion,
    - String fileName,
    - String fileTypeCode

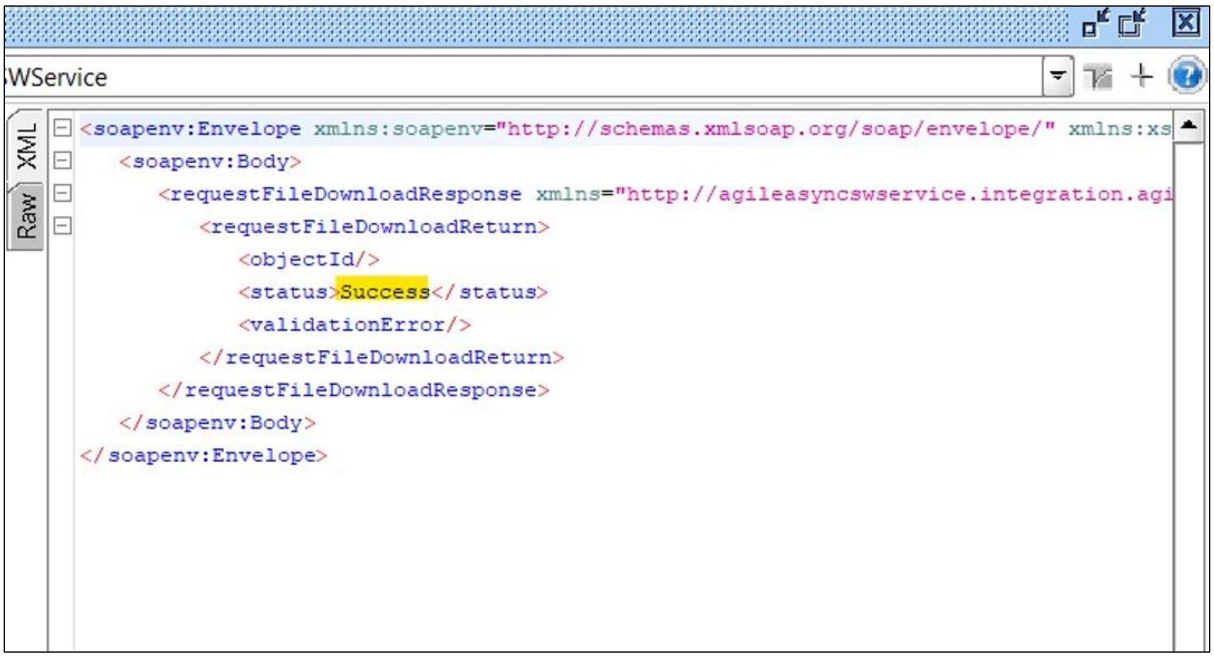
- Response:
    - Response will content status Success/Failure along with ObjectID and Validation Error.

- Example:
    - Request from Downstream


Request 1
https://agilereporting.us.dell.com:7002/AgileAsyncSWService/services/AgileAsyn
XML
Raw

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:agil=
   <soapenv:Header/>
   <soapenv:Body>
      <agil:requestFileDownload>
         <agil:messageID>183</agil:messageID>
         <agil:downstreamName>CPG</agil:downstreamName>
         <agil:downStreamUserName>aswath_V</agil:downStreamUserName>
         <agil:swb_Number>02M4H</agil:swb_Number>
         <agil:swb_Rev>X05-00</agil:swb_Rev>
         <agil:folderNumber>FOLDER11267035M</agil:folderNumber>
         <agil:folderVersion>1</agil:folderVersion>
         <agil:fileName>ESXiAdditionalAttributes.xml</agil:fileName>
         <agil:fileTypeCode> XML:An XML file</agil:fileTypeCode>
      </agil:requestFileDownload>
   </soapenv:Body>
</soapenv:Envelope>
```

- Response from Agile



```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xs
   <soapenv:Body>
      <requestFileDownloadResponse xmlns="http://agileasyncswservice.integration.agi
         <requestFileDownloadReturn>
            <objectId/>
            <status>Success</status>
            <validationError/>
         </requestFileDownloadReturn>
      </requestFileDownloadResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

requestRCDFileDownload

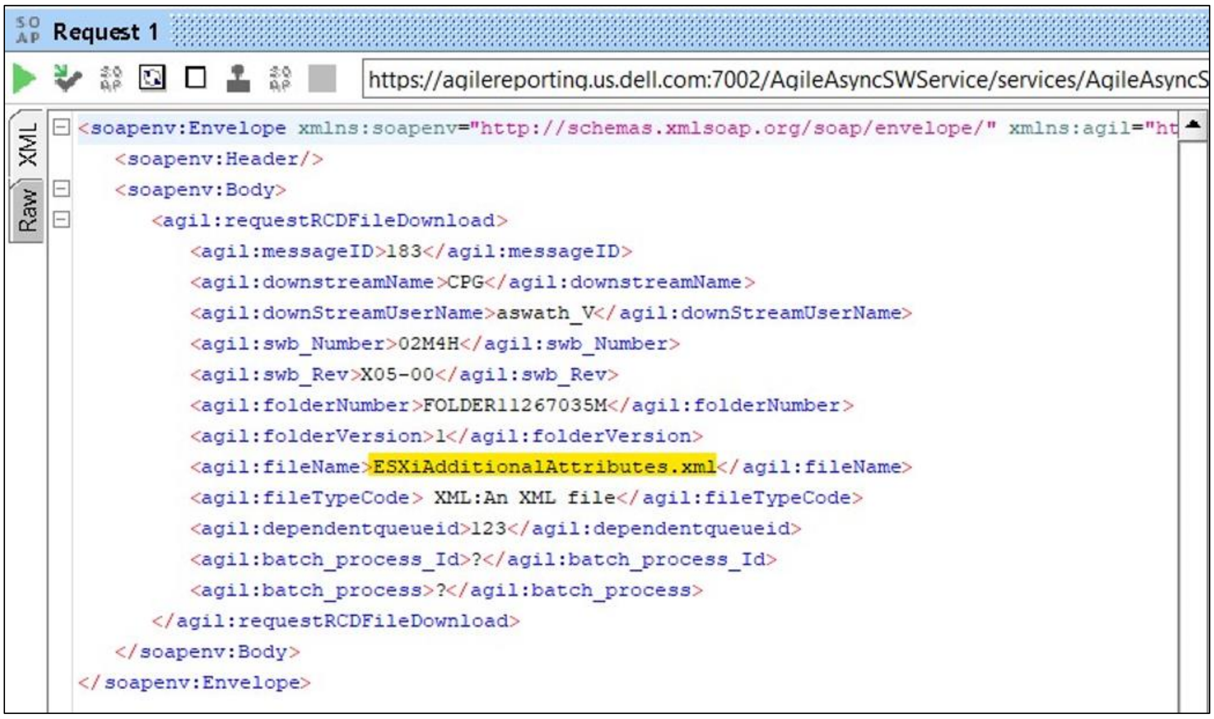
- Method Description:
  - The method processes a request for downloading a file by performing several validation checks, interacting with a database to execute a stored procedure, and handling any exceptions that occur. It ensures that all necessary parameters are provided and valid before proceeding with the database operations.

- Working:
  - requestRCDFileDownload takes the below parameters, does all the validation and download process is performed in this method.
  - AddServiceQueueDAO dao object is created and call to method addServiceQueue.
  - addServiceQueue method takes all the parameters process it to create a query and execute to enter a record into the database.

- Parameters:
  - String messageID,
  - String downstreamName,
  - String downStreamUserName,
  - String swb_Number,
  - String swb_Rev,
  - String folderNumber,
  - String folderVersion,
  - String fileName,
  - String fileTypeCode,
  - int dependentqueueid,
  - String batch_process_Id,
  - String batch_process

- Response:
  - Response will content status Success/Failure along with ObjectID and Validation Error.

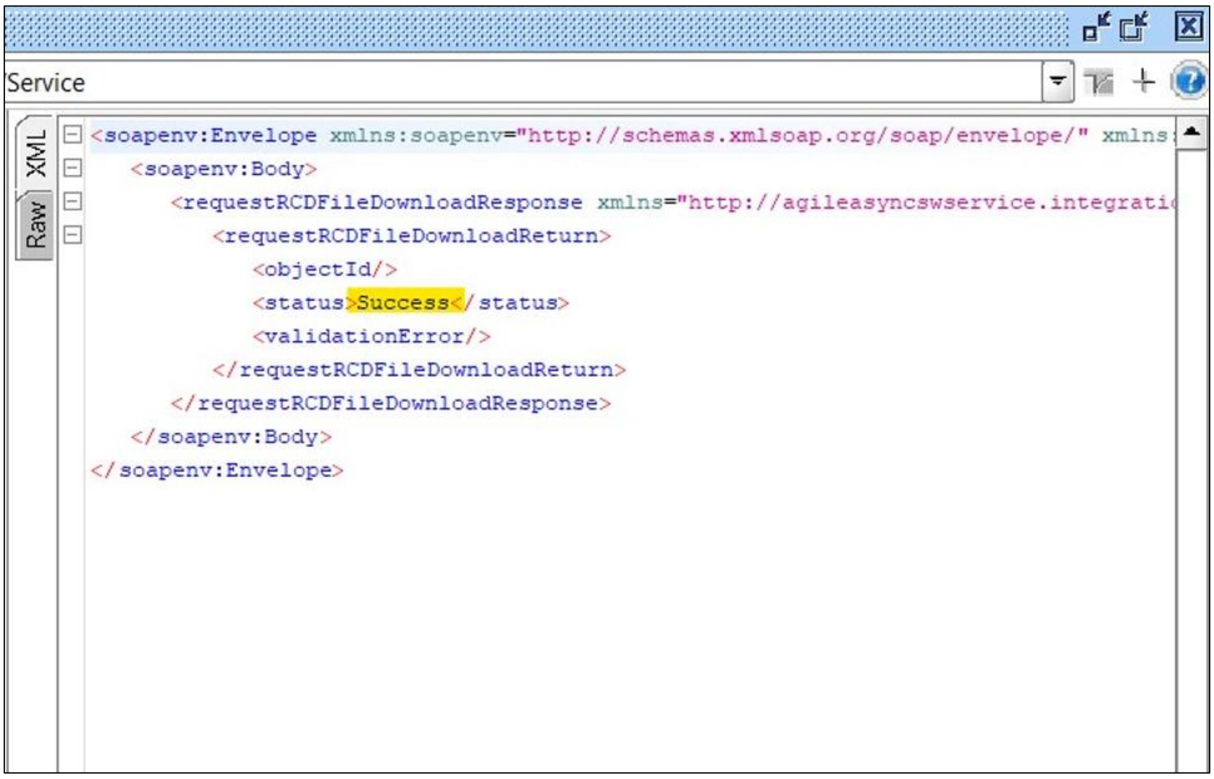
- Example:
  - Request from Downstream



```
https://agilereporting.us.dell.com:7002/AgileAsyncSWService/services/AgileAsyncS
```

```xml
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:agil="ht
   <soapenv:Header/>
   <soapenv:Body>
      <agil:requestRCDFileDownload>
         <agil:messageID>183</agil:messageID>
         <agil:downstreamName>CPG</agil:downstreamName>
         <agil:downStreamUserName>aswath_V</agil:downStreamUserName>
         <agil:swb_Number>02M4H</agil:swb_Number>
         <agil:swb_Rev>X05-00</agil:swb_Rev>
         <agil:folderNumber>FOLDER11267035M</agil:folderNumber>
         <agil:folderVersion>1</agil:folderVersion>
         <agil:fileName>ESXiAdditionalAttributes.xml</agil:fileName>
         <agil:fileTypeCode> XML:An XML file</agil:fileTypeCode>
         <agil:dependentqueueid>123</agil:dependentqueueid>
         <agil:batch_process_Id>?</agil:batch_process_Id>
         <agil:batch_process>?</agil:batch_process>
      </agil:requestRCDFileDownload>
   </soapenv:Body>
</soapenv:Envelope>
```

  - Response from Agile



```xml
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:
   <soapenv:Body>
      <requestRCDFileDownloadResponse xmlns="http://agileasyncswservice.integrati
         <requestRCDFileDownloadReturn>
            <objectId/>
            <status>Success</status>
            <validationError/>
         </requestRCDFileDownloadReturn>
      </requestRCDFileDownloadResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

requestRCDDownloadAck

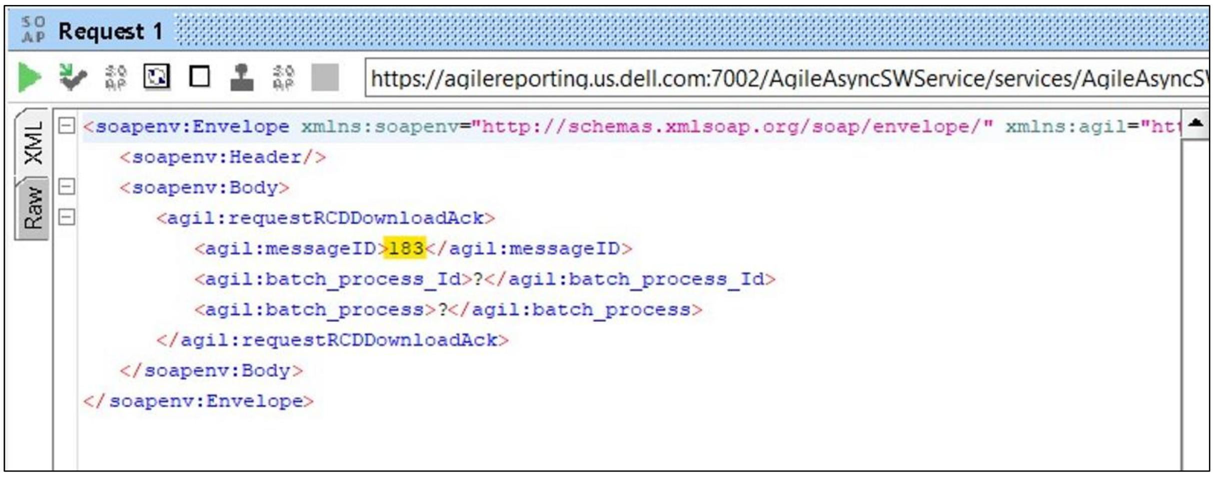
- Method Description:
  - This method is used as a download acknowledgement.

- Working:
  - This method processes an acknowledgment request for an RCD file download by performing validation checks, interacting with a database to execute a stored procedure, and handling any exceptions that occur.
  - It ensures that all necessary parameters are provided and valid before proceeding with the database operations.
  - AddServiceQueueDAO dao object is created and call to method addServiceQueue.
  - addServiceQueue method takes all the parameters process it to create a query and execute to enter a record into the database.

- Parameters:
  - String messageID,
  - String batch_process_Id,
  - String batch_process

- Response:
  - Response will content status Success/Failure along with ObjectID and Validation Error.

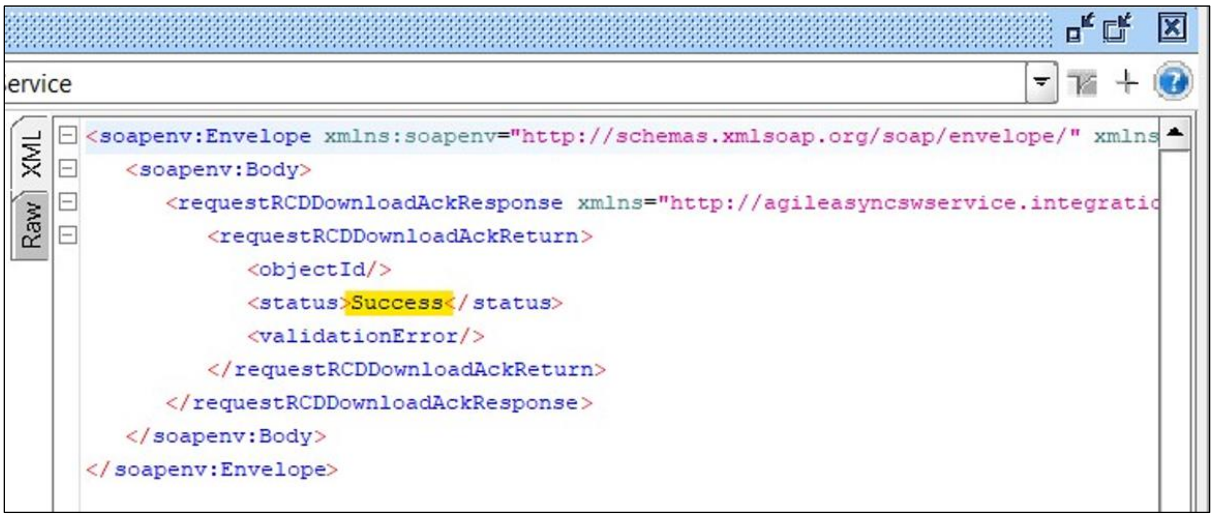
- Example:
  - Request from Downstream



https://agilereporting.us.dell.com:7002/AgileAsyncSWService/services/AgileAsyncS

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:agil="ht
   <soapenv:Header/>
   <soapenv:Body>
      <agil:requestRCDDownloadAck>
         <agil:messageID>183</agil:messageID>
         <agil:batch_process_Id>?</agil:batch_process_Id>
         <agil:batch_process>?</agil:batch_process>
      </agil:requestRCDDownloadAck>
   </soapenv:Body>
</soapenv:Envelope>
```

- Response from Agile



```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns
   <soapenv:Body>
      <requestRCDDownloadAckResponse xmlns="http://agileasyncswservice.integratic
         <requestRCDDownloadAckReturn>
            <objectId/>
            <status>Success</status>
            <validationError/>
         </requestRCDDownloadAckReturn>
      </requestRCDDownloadAckResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

createAutoReGenerationWPCO

- Method Description:
  - The method is responsible for creating an automatic generation of a WPCO (Web Processing Change Order) based on the provided downstream name and swb parameters. It validates input parameters, create an Agile session, processing WPCO auto-regeneration.

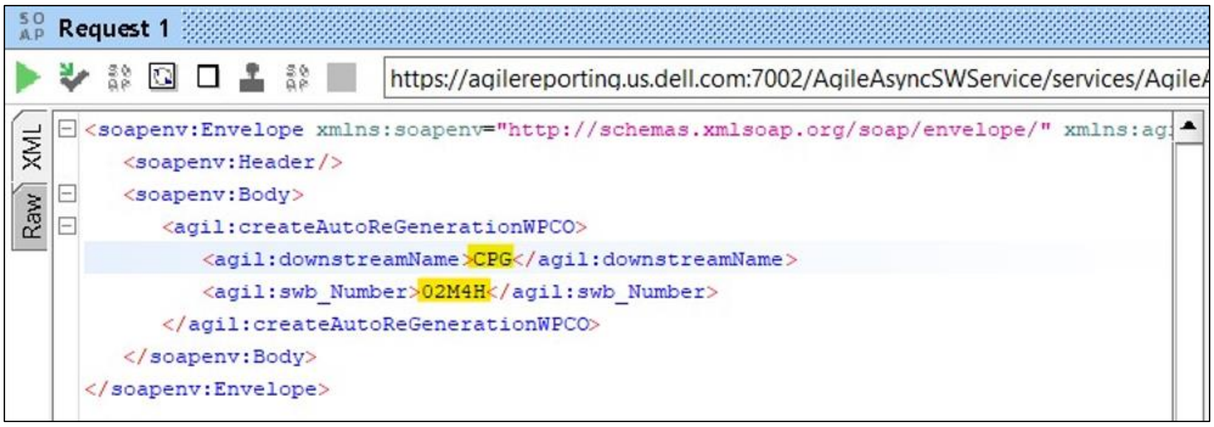
- Working:
  - Checks if downstream name is empty and appends an error message if true.
  - Validates the downstream name against a list of valid names using a utility method.

- Checks if swb number is empty and appends an error message if true.
- Create Agile Session
- Validate SWB
- Process Auto-ReGeneration
- Response Success or Failure.

➢ Parameters:
  - String downstreamName
  - String swb_Number

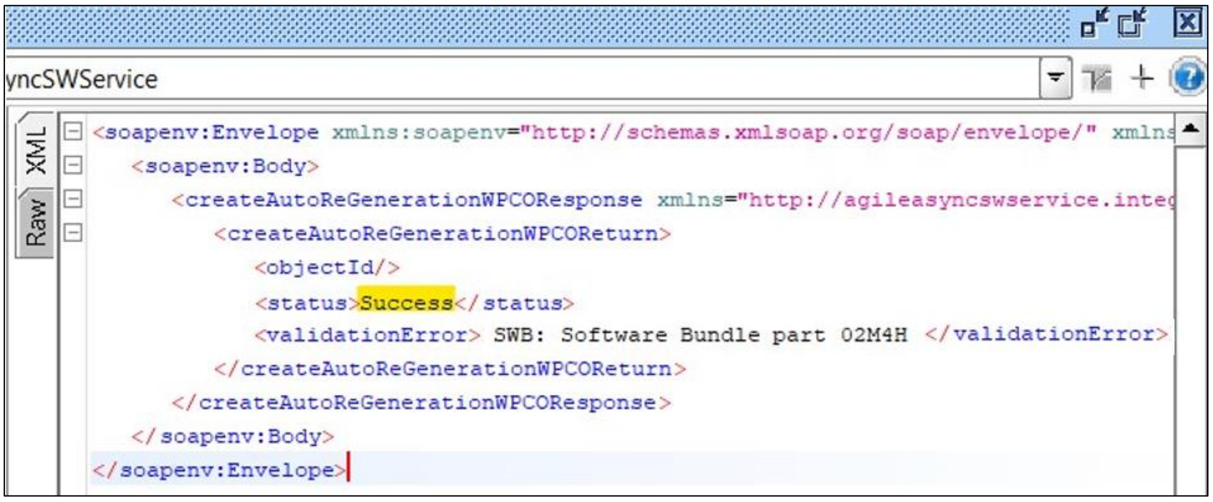
➢ Response:
  - Response will content status Success/Failure along with ObjectID and Validation Error.

➢ Example:
  - Request from Downstream


Request 1
https://agilereporting.us.dell.com:7002/AgileAsyncSWService/services/AgileA
XML
Raw

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:ag
   <soapenv:Header/>
   <soapenv:Body>
      <agil:createAutoReGenerationWPCO>
         <agil:downstreamName>CPG</agil:downstreamName>
         <agil:swb_Number>02M4H</agil:swb_Number>
      </agil:createAutoReGenerationWPCO>
   </soapenv:Body>
</soapenv:Envelope>
```

  - Response from Agile



```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns
   <soapenv:Body>
      <createAutoReGenerationWPCOResponse xmlns="http://agileasyncswservice.integ
         <createAutoReGenerationWPCOReturn>
            <objectId/>
            <status>Success</status>
            <validationError> SWB: Software Bundle part 02M4H </validationError>
         </createAutoReGenerationWPCOReturn>
      </createAutoReGenerationWPCOResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

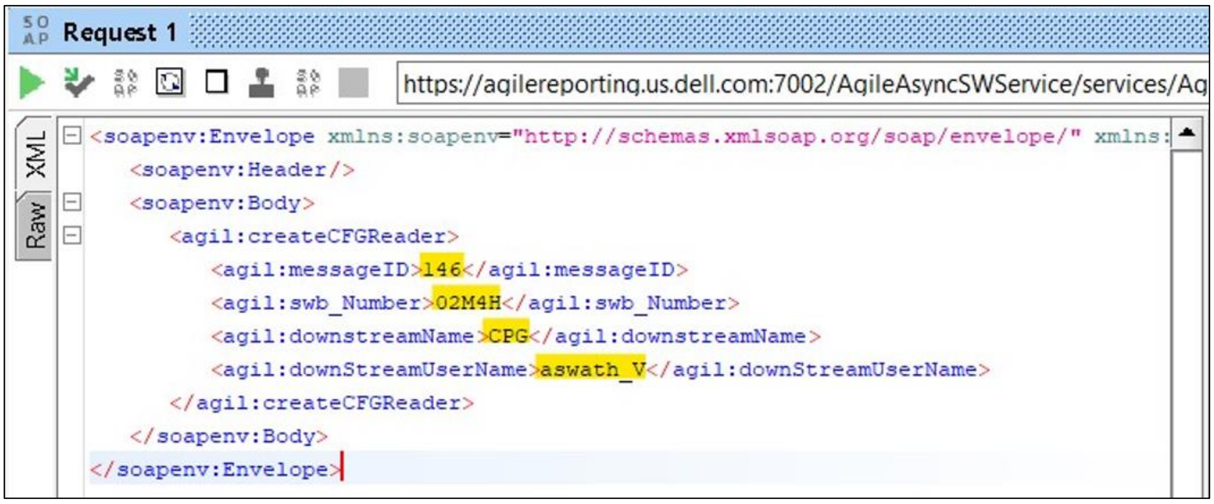
createCFGReader

- Method Description:
    - This method could be used in a scenario where a service needs to validate input parameters and interact with a database to execute a stored procedure, encapsulating the result in a Response object.

- Working:
    - Configure Logging
    - Validates the messageID, swb_Number, downstreamName, downStreamUserName.
    - Validate Downstream Name
    - Database connection established
    - Called addServiceQueue method
    - Response Success or Failure

- Parameters:
    - messageID
    - swb_Number
    - downstreamName
    - downStreamUserName

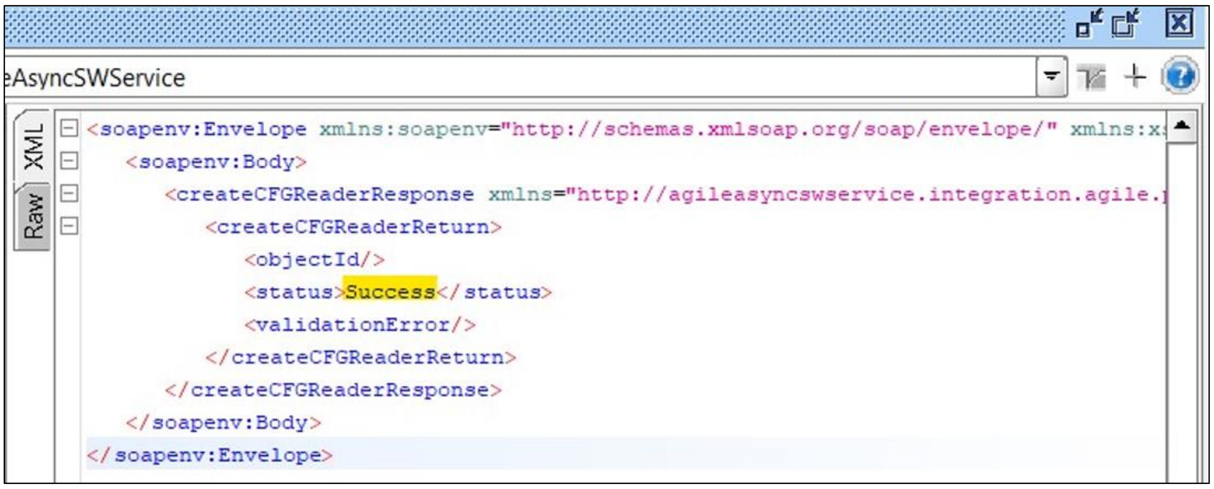
- Response:
    - Response will content status Success/Failure along with ObjectID and Validation Error.
- Example:
    - Request from Downstream



```
https://agilereporting.us.dell.com:7002/AgileAsyncSWService/services/Ag
```

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:
   <soapenv:Header/>
   <soapenv:Body>
      <agil:createCFGReader>
         <agil:messageID>146</agil:messageID>
         <agil:swb_Number>02M4H</agil:swb_Number>
         <agil:downstreamName>CPG</agil:downstreamName>
         <agil:downStreamUserName>aswath_V</agil:downStreamUserName>
      </agil:createCFGReader>
   </soapenv:Body>
</soapenv:Envelope>
```

- Response from Agile



```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:x
   <soapenv:Body>
      <createCFGReaderResponse xmlns="http://agileasyncswservice.integration.agile.
         <createCFGReaderReturn>
            <objectId/>
            <status>Success</status>
            <validationError/>
         </createCFGReaderReturn>
      </createCFGReaderResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

createDSCR

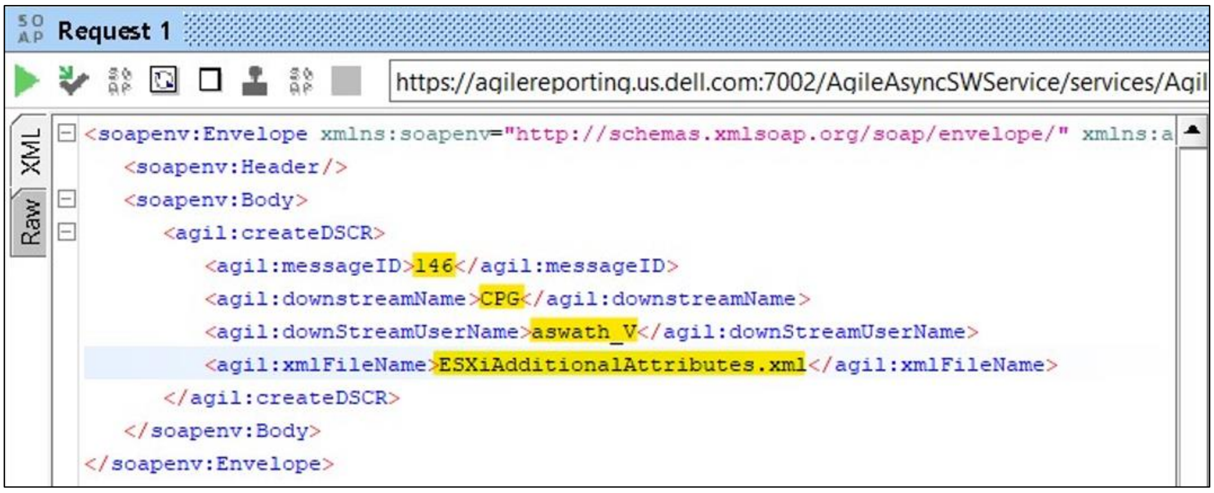
- Method Description:
  - The createDSCR method validates inputs, loads properties, validates downstream names, creates a DSCR number, interacts with the database to insert a record, handles exceptions, and constructs a response object indicating the result of the operation.

- Working:
  - Configure Logging
  - Validate if messageID, swb_Number, downstreamName, downStreamUserName are empty
  - Validate Downstream Name
  - Create Agile Session
  - Database Connection
  - Called addServiceQueue method
  - Response Success or Failure

- Parameters:
  - messageID
  - downstreamName
  - downStreamUserName
  - xmlFileName

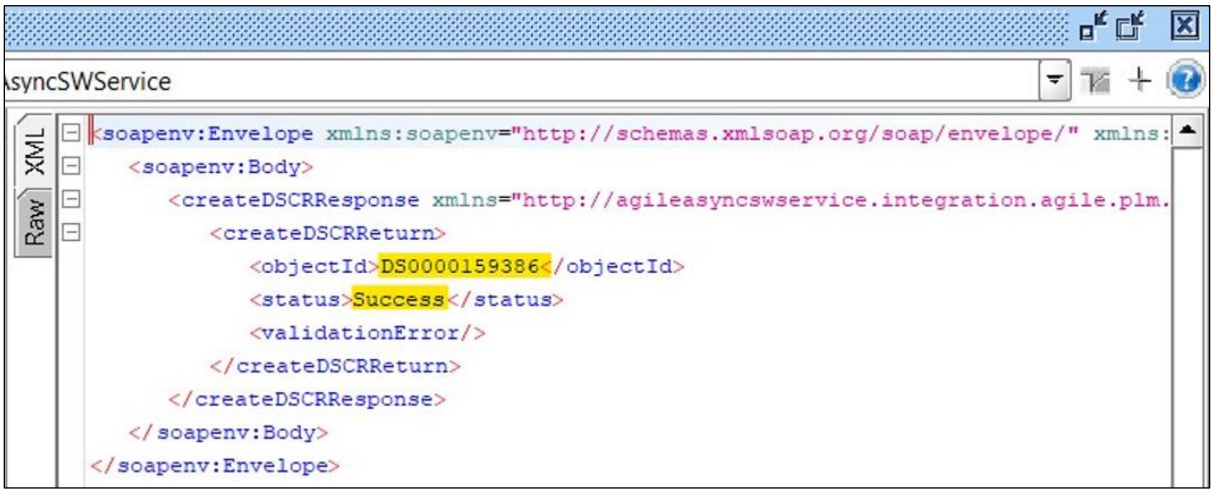
- Response:
  - Response will content status Success/Failure along with ObjectID and Validation Error.

- Example:
  - Request from Downstream



```
https://agilereporting.us.dell.com:7002/AgileAsyncSWService/services/Agil
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:a
   <soapenv:Header/>
   <soapenv:Body>
      <agil:createDSCR>
         <agil:messageID>146</agil:messageID>
         <agil:downstreamName>CPG</agil:downstreamName>
         <agil:downStreamUserName>aswath_V</agil:downStreamUserName>
         <agil:xmlFileName>ESXiAdditionalAttributes.xml</agil:xmlFileName>
      </agil:createDSCR>
   </soapenv:Body>
</soapenv:Envelope>
```

  - Response from Agile



```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:
   <soapenv:Body>
      <createDSCRResponse xmlns="http://agileasyncswservice.integration.agile.plm.
         <createDSCRReturn>
            <objectId>DS0000159386</objectId>
            <status>Success</status>
            <validationError/>
         </createDSCRReturn>
      </createDSCRResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

createPCID

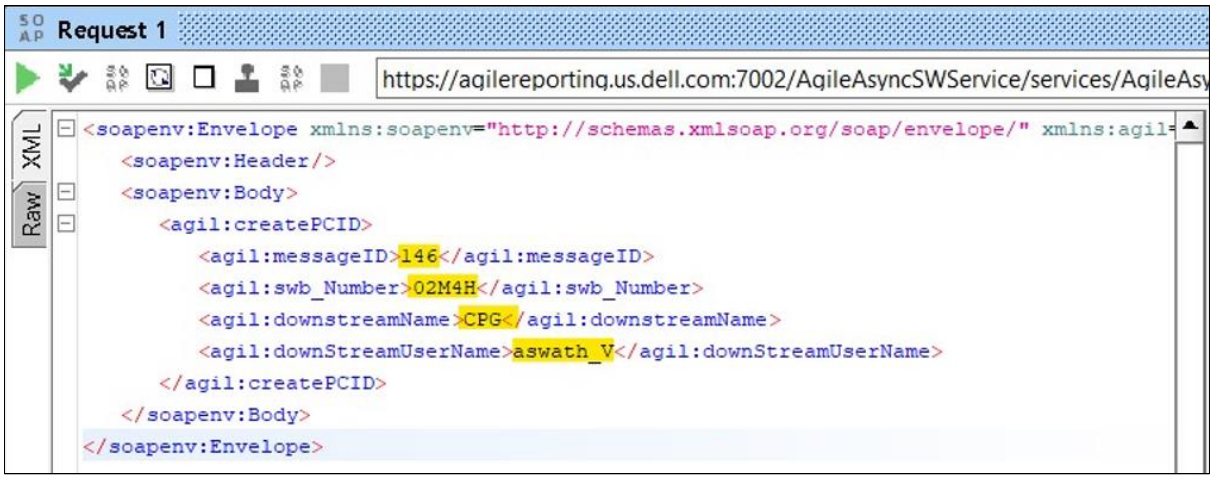
- Method Description:
  - The createPCID method is designed to handle the creation of a PCID by validating input parameters, interacting with a database, and managing exceptions.

- Working:
  - Configure Logging
  - Validate if messageID, swb_Number, downstreamName, downStreamUserName are empty
  - Validate Downstream Name
  - Database Connection
  - Call addServiceQueue method
  - Response Success or Failure

- Parameters:
  - messageID
  - swb_Number
  - downstreamName
  - downStreamUserName

- Response:
  - Response will content status Success/Failure along with ObjectID and Validation Error.

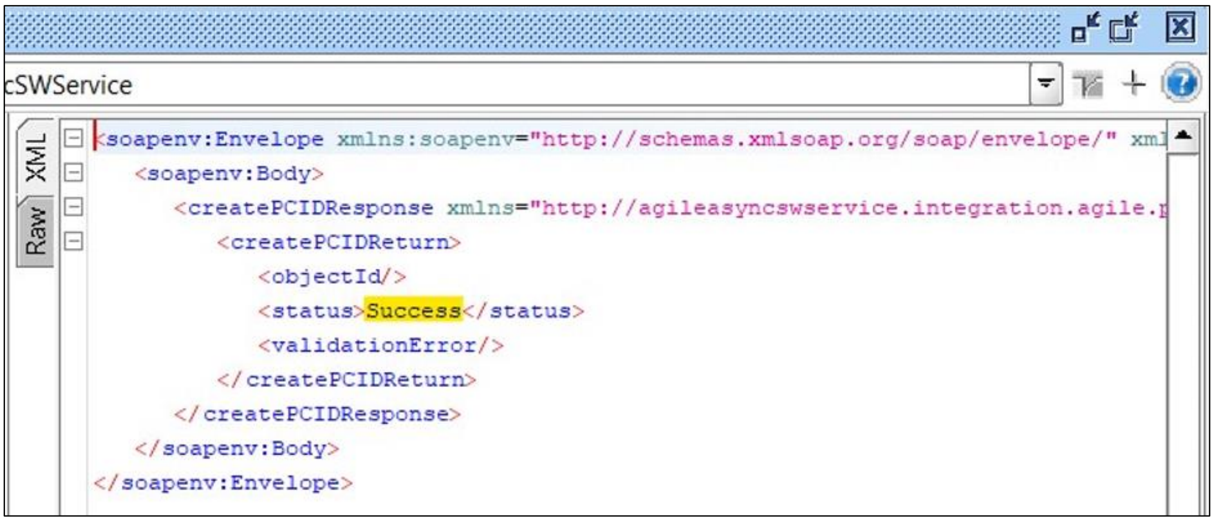
- Example:
  - Request from Downstream



https://agilereporting.us.dell.com:7002/AgileAsyncSWService/services/AgileAsy

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:agil=
   <soapenv:Header/>
   <soapenv:Body>
      <agil:createPCID>
         <agil:messageID>146</agil:messageID>
         <agil:swb_Number>02M4H</agil:swb_Number>
         <agil:downstreamName>CPG</agil:downstreamName>
         <agil:downStreamUserName>aswath_V</agil:downStreamUserName>
      </agil:createPCID>
   </soapenv:Body>
</soapenv:Envelope>
```

- Response from Agile


cSWService

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xml
   <soapenv:Body>
      <createPCIDResponse xmlns="http://agileasyncswservice.integration.agile.p
         <createPCIDReturn>
            <objectId/>
            <status>Success</status>
            <validationError/>
         </createPCIDReturn>
      </createPCIDResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

createRCDBuild

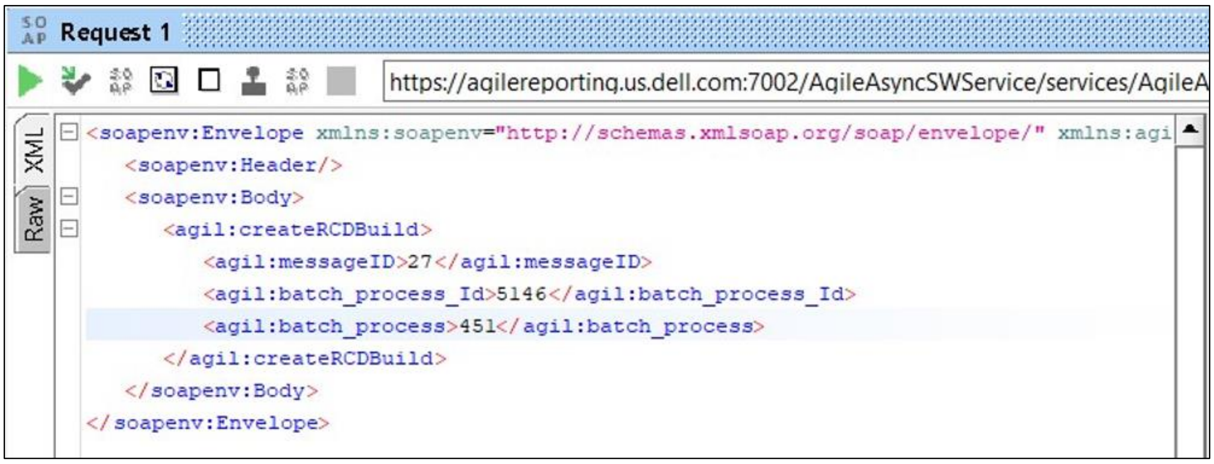
- Method Description:
  - The createRCDBuild method is designed to handle the creation, validate input parameters, interact with a database, and manage exceptions.

- Working:
  - Configure Logging
  - Validates the messageID, batch_process_Id and batch_process
  - Database connection established
  - Called addServiceQueue method
  - Response Success or Failure

- Parameters:
  - messageID
  - batch_process_Id
  - batch_process

- Response:
  - Response will content status Success/Failure along with ObjectID and Validation Error.

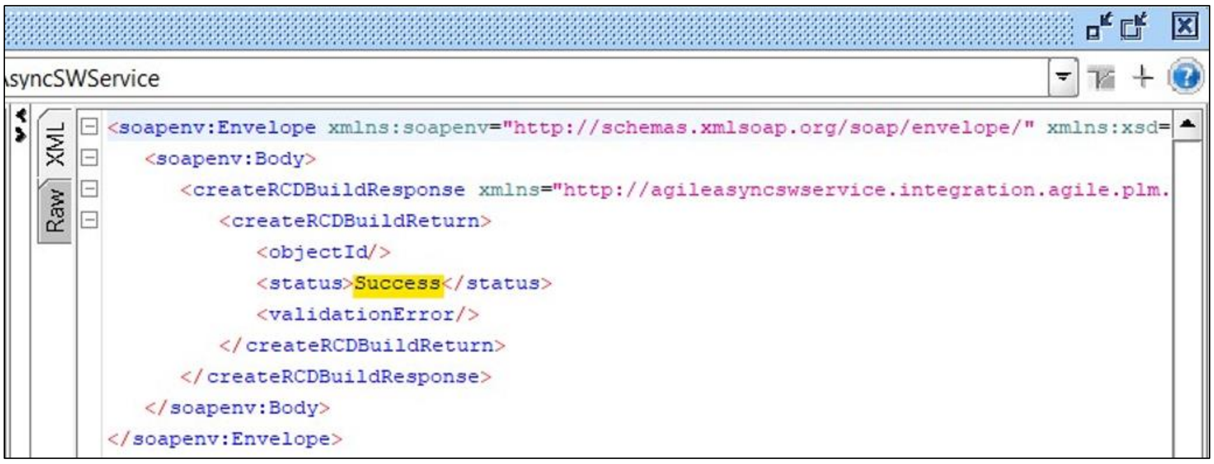
➢ Example:
  - Request from Downstream



https://agilereporting.us.dell.com:7002/AgileAsyncSWService/services/AgileA

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:agi
   <soapenv:Header/>
   <soapenv:Body>
      <agil:createRCDBuild>
         <agil:messageID>27</agil:messageID>
         <agil:batch_process_Id>5146</agil:batch_process_Id>
         <agil:batch_process>451</agil:batch_process>
      </agil:createRCDBuild>
   </soapenv:Body>
</soapenv:Envelope>
```

  - Response from Agile



```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsd=
   <soapenv:Body>
      <createRCDBuildResponse xmlns="http://agileasyncswservice.integration.agile.plm.
         <createRCDBuildReturn>
            <objectId/>
            <status>Success</status>
            <validationError/>
         </createRCDBuildReturn>
      </createRCDBuildResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

ftpPush

➢ Method Description:
  - The ftpPush method is designed to handle FTP (File Transfer Protocol) operations by validating input parameters, executing a database stored procedure, and managing exceptions.

➢ Working:
  - Configure Logging
  - Extract Values from folderDetailsRequest
  - Validates the messageID, partNumber and folderDetails

- Folder Details Validation; folderNumber, fileName, fileTypeCode, folderVersion
- Database connection established
- Called addServiceQueue method
- Execute the stored procedure for each FolderDetails object
- Clean up resources in the finally block.
- Response Success or Failure

➢ Parameters:

- folderDetailsRequest

➢ Response:

- Response will content status Success/Failure along with ObjectID and Validation Error.

getDUPFileStatus

➢ Method Description:

- The getDUPFileStatus method retrieves the status of a file based on the provided messageID, downstreamName, and swb_Number. It validates the inputs, interacts with a database to execute a stored procedure, and constructs a response object with the status details.

➢ Working:

- Configure Logging
- Validates the messageID, downstreamName and swb_Number
- Instantiate ServicePLMAppUserDAO and establish a database connection
- Execute the stored procedure "{call AIC_FILETYPECODES_PKG.PRC_GETFILETYPEVALUES(messageID,swb_Number,OracleTypes.CURSOR)}" and process the result set.
- Set Successful Response
- Clean up resources in the finally block.
- Response Success or Failure

➢ Parameters:

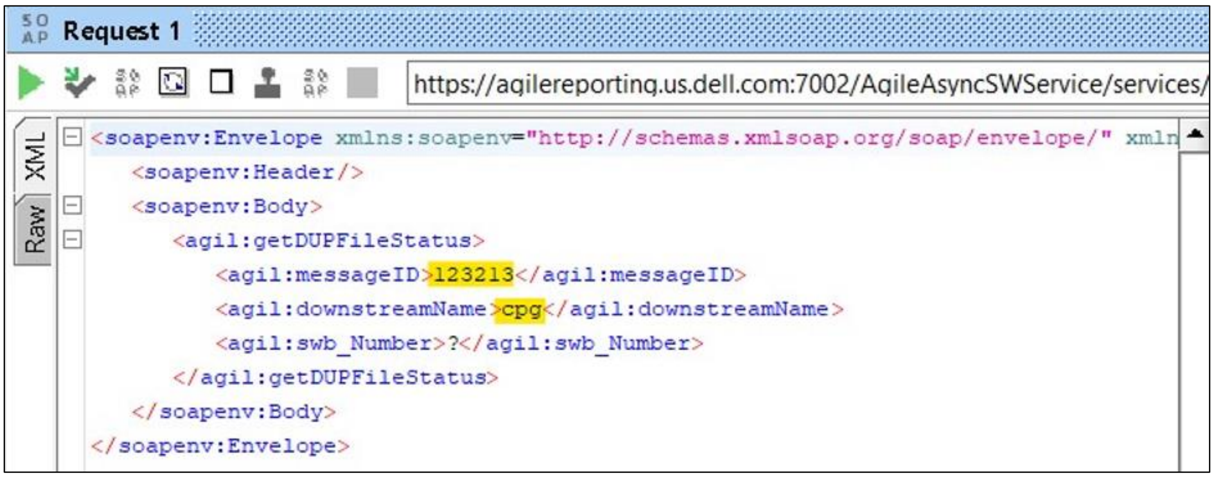
- messageID
- downstreamName
- swb_Number

➢ Response:

- Response will content status Success/Failure along with ObjectID and Validation Error.

➢ Example:

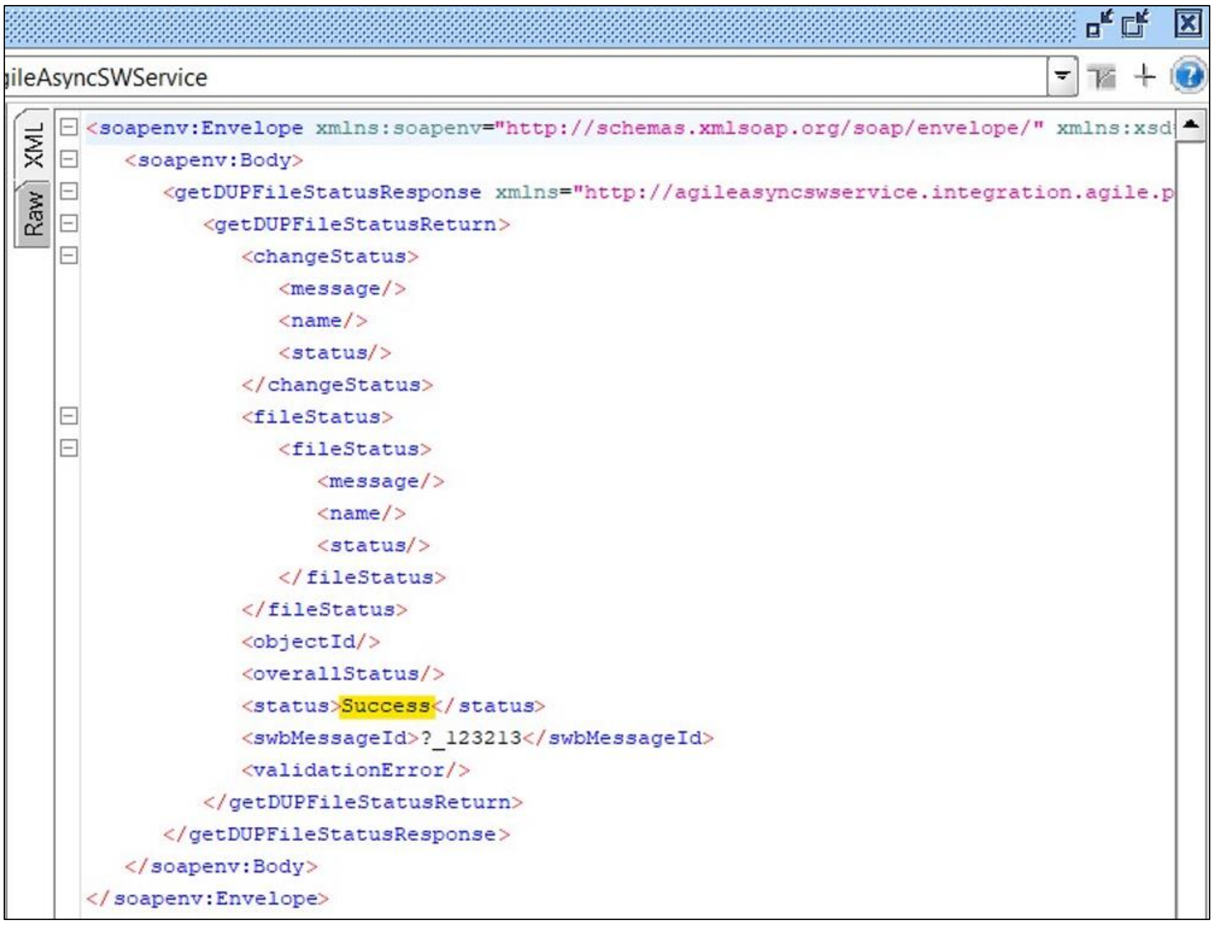
- Request from Downstream



https://agilereporting.us.dell.com:7002/AgileAsyncSWService/services/

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmln
   <soapenv:Header/>
   <soapenv:Body>
      <agil:getDUPFileStatus>
         <agil:messageID>123213</agil:messageID>
         <agil:downstreamName>cpg</agil:downstreamName>
         <agil:swb_Number>?</agil:swb_Number>
      </agil:getDUPFileStatus>
   </soapenv:Body>
</soapenv:Envelope>
```

- Response from Agile



```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsd
   <soapenv:Body>
      <getDUPFileStatusResponse xmlns="http://agileasyncswservice.integration.agile.p
         <getDUPFileStatusReturn>
            <changeStatus>
               <message/>
               <name/>
               <status/>
            </changeStatus>
            <fileStatus>
               <fileStatus>
                  <message/>
                  <name/>
                  <status/>
               </fileStatus>
            </fileStatus>
            <objectId/>
            <overallStatus/>
            <status>Success</status>
            <swbMessageId>?_123213</swbMessageId>
            <validationError/>
         </getDUPFileStatusReturn>
      </getDUPFileStatusResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

getProcessStatus

- Method Description:
  - The getProcessStatus method retrieves the processing status of a given message from a database, based on the provided messageId, downStreamName, and objectReferenceNum. It interacts with a stored procedure to get the required information, processes the result set, and constructs a response object with the status details.

- Working:
  - Configure Logging
  - Load application properties and SWB properties
  - Validates the messageID, and downStreamName
  - Database connection established
  - Called addServiceQueue method
  - Execute the stored procedure "{call AIC_FILETYPECODES_PKG.PRC_GETFILETYPEVALUES(messageID,down

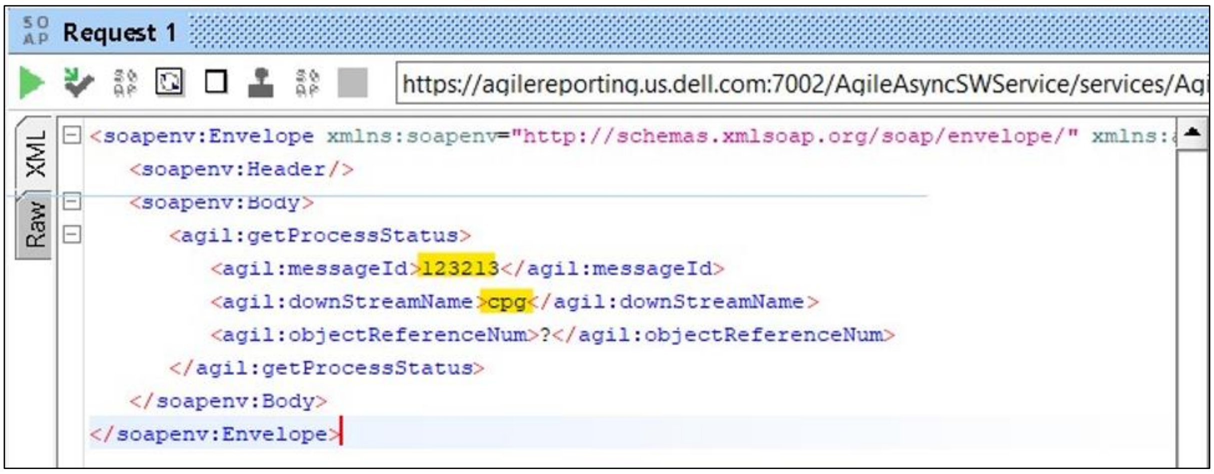
StreamName,objectReferenceNum, OracleTypes.CURSOR)}" and process the result set.

- Clean up resources in the finally block.
- Response Success or Failure.

➢ Parameters:
  - messageID
  - downstreamName
  - objectReferenceNum

➢ Response:
  - Response will content status Success/Failure along with ObjectID and Validation Error.

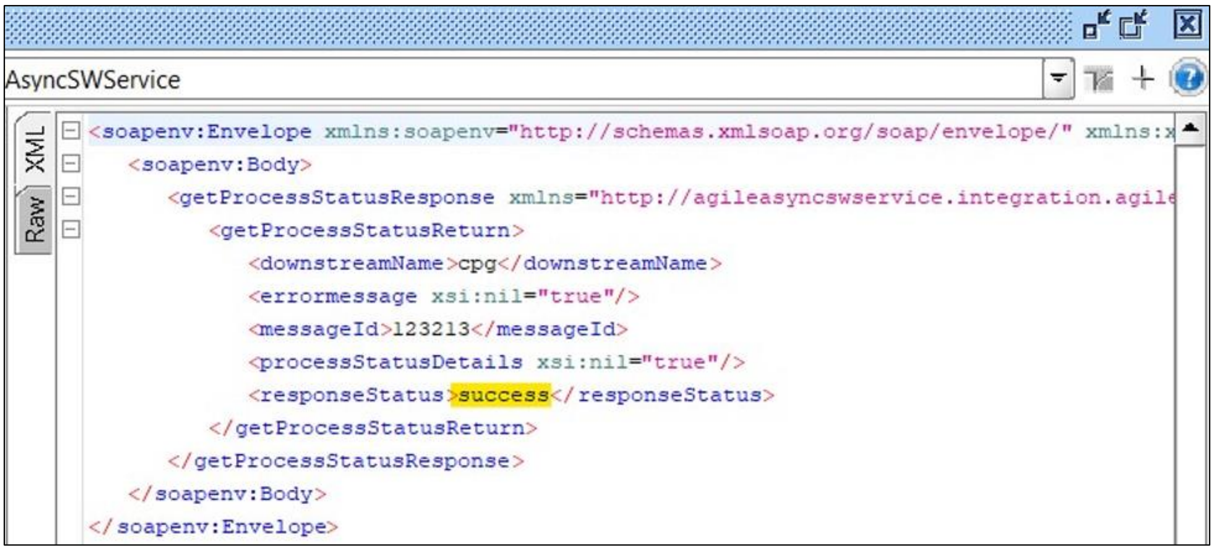
➢ Example:
  - Request from Downstream



https://agilereporting.us.dell.com:7002/AgileAsyncSWService/services/Agi

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:
   <soapenv:Header/>
   <soapenv:Body>
      <agil:getProcessStatus>
         <agil:messageId>123213</agil:messageId>
         <agil:downStreamName>cpg</agil:downStreamName>
         <agil:objectReferenceNum>?</agil:objectReferenceNum>
      </agil:getProcessStatus>
   </soapenv:Body>
</soapenv:Envelope>
```

  - Response from Agile



```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:x
   <soapenv:Body>
      <getProcessStatusResponse xmlns="http://agileasyncswservice.integration.agile
         <getProcessStatusReturn>
            <downstreamName>cpg</downstreamName>
            <errormessage xsi:nil="true"/>
            <messageId>123213</messageId>
            <processStatusDetails xsi:nil="true"/>
            <responseStatus>success</responseStatus>
         </getProcessStatusReturn>
      </getProcessStatusResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

processEsupportAck

➢ Method Description:

- The processEsupportAck method handles the acknowledgment process for e-support systems. It validates the input parameters, ensures that the downstream system is authorized, creates an Agile session, retrieves an Agile object, and processes the acknowledgment.

➢ Working:

- Configure Logging
- Load application properties and standard system configuration properties.
- Validates downStreamName, storeId, agileIdentifier
- Create Agile Session
- Process Acknowledgment create ProcessEsupportAckBO object
- Call doAckProcessing method.
- Clean up resources in the finally block.
- Response Success or Failure

➢ Parameters:

- agileIdentifier
- storeId
- downStreamName

- Response:
  - Response will content status Success/Failure.

processSWBtoXRev

- Method Description:
  - The processSWBtoXRev method handles the creation of PNRs based on a request. It validates the request attributes, creates an Agile session, and processes the PNR creation.

- Working:
  - Configure Logging
  - Loads application and SWB properties.
  - Validates mandatory attributes of the request.
  - Loads standard system configuration properties.
  - Creates an Agile session.
  - Processes the PNR creation.
  - Handles exceptions and ensures the Agile session is closed properly.
  - Returns a response object with the status and details of the PNR creation process

- Parameters:
  - pnrCreationRequest

- Response:
  - Response will content status Success/Failure.

psrFileDownload

- Method Description:
  - The psrFileDownload method facilitates the process of downloading a PSR file. It performs validation, initiates a connection to an Agile session, executes a stored procedure to handle the download request, and manages the response.

- Working:
    - Configures logging and loads necessary properties.
    - Creates an Agile session to interact with the Agile system.
    - Validates input parameters to ensure all mandatory fields are provided and checks if the current user is authorized.
    - Performs PSR type validation using a validation utility.
    - Executes a stored procedure to handle the download request.
    - Handles exceptions and updates the response object accordingly.
    - Cleans up resources to close connections and sessions properly.

- Parameters:
    - messageID
    - downstreamName
    - psr_Number
    - fileName
    - fileExtension

- Response:
    - Response will content status Success/Failure.

requestCPGFileAction

- Method Description:
    - The requestCPGFileAction method is designed to handle requests for CPG file actions. This method involves performing basic and file-specific validations, managing Agile sessions, and generating appropriate responses.

- Working:
    - Configures logging and loads necessary properties.
    - Performs basic validations using the CPGValidations service to ensure that all required input parameters are valid.
    - Creates an Agile session if basic validations pass.
    - Performs file-specific validations using the Agile session and the CPGValidations service.

- Sets the response status to "Success" if all validations pass.
- Handles exceptions and updates the response object accordingly.
- Cleans up resources to close connections and sessions properly.

➢ Parameters:

- messageID
- downstreamName
- partNum
- fileRequest

➢ Response:

- Response will content status Success/Failure.

requestDFFileDownload

➢ Method Description:

- The requestDFFileDownload method processes requests for downloading DF (Design Files) files. This involves validating the request, inserting records into a queue, and handling database connections.

➢ Working:

- Configures logging and loads necessary properties.
- Validates the request to ensure that the message ID and part details are provided.
- Inserts records into the queue using stored procedures.
- Handles exceptions and updates the response object accordingly.
- Cleans up resources to close connections and statements properly.

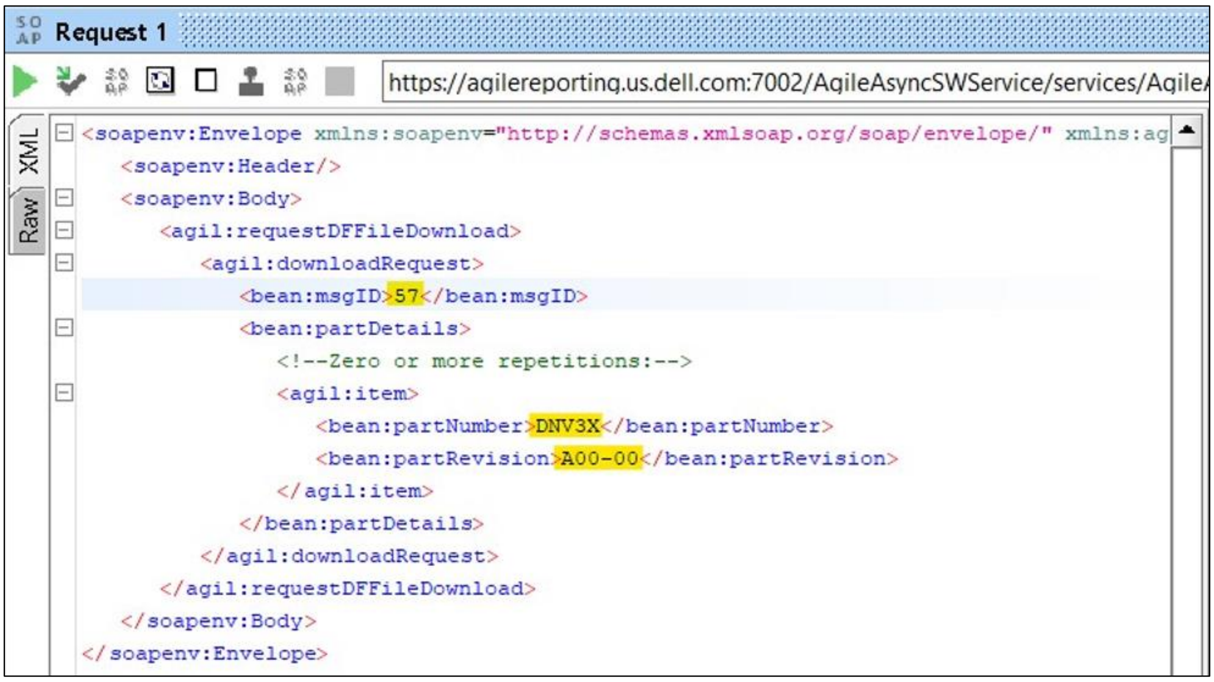
➢ Parameters:

- downloadRequest

➢ Response:

- Response will content status Success/Failure.

- Example:
  - Request from Downstream

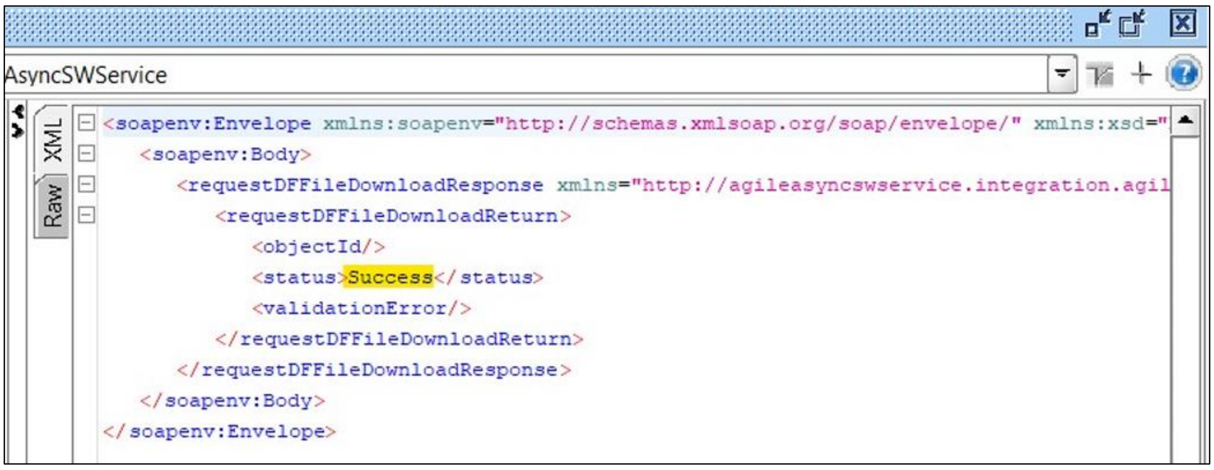


Request 1

https://agilereporting.us.dell.com:7002/AgileAsyncSWService/services/AgileA

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:ag
   <soapenv:Header/>
   <soapenv:Body>
      <agil:requestDFFileDownload>
         <agil:downloadRequest>
            <bean:msgID>57</bean:msgID>
            <bean:partDetails>
               <!--Zero or more repetitions:-->
               <agil:item>
                  <bean:partNumber>DNV3X</bean:partNumber>
                  <bean:partRevision>A00-00</bean:partRevision>
               </agil:item>
            </bean:partDetails>
         </agil:downloadRequest>
      </agil:requestDFFileDownload>
   </soapenv:Body>
</soapenv:Envelope>
```

  - Response from Agile



AsyncSWService

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsd="
   <soapenv:Body>
      <requestDFFileDownloadResponse xmlns="http://agileasyncswservice.integration.agil
         <requestDFFileDownloadReturn>
            <objectId/>
            <status>Success</status>
            <validationError/>
         </requestDFFileDownloadReturn>
      </requestDFFileDownloadResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

requestSRVDetailsForSWB

- Method Description:
  - The requestSRVDetailsForSWB method handles requests for retrieving SRV details for a SWB. The method performs parameter validation, database interactions, and ensures proper resource management.

- Working:
  - Configures logging and initializes local variables.
  - Validates the input parameters to ensure they are not empty and the downstream name is valid.
  - Handles validation errors by throwing a Validation Exception.
  - Interacts with the database to insert the request details into a service queue.
  - Handles exceptions and updates the response object accordingly.
  - Cleans up resources to close connections and statements properly.

- Parameters:
  - messageID
  - downstreamName
  - downStreamUserName
  - swb_Number

- Response:
  - Response will content status Success/Failure.

requestUtilityFileAction

- Method Description:
  - The requestUtilityFileAction method handles requests for file actions related to utility files. This method performs validation, interacts with an Agile session, and manages resources efficiently.

- Working:
  - Configures logging and initializes local variables.
  - Loads application properties, Agile PC references, and system configuration properties.
  - Performs basic validations on the input parameters.
  - Creates an Agile session and performs file data validations.
  - Sets the response status to success if validations pass.
  - Handles exceptions and updates the response object accordingly.

    - Cleans up resources to close connections and release memory properly.

- Parameters:
    - messageID
    - downstreamName
    - swb_Number
    - swb_Rev
    - fileDetailsRequest
    - autoRegeneration

- Response:
    - Response will content status Success/Failure.

structureSWBtoParentParts

- Method Description:
    - The structureSWBtoParentParts method processes a request to structure SWB (Software Bill) details to parent parts, validates the request, writes the request to a JSON file, and inserts details into a queue.

- Working:
    - Configures logging and initializes the response object.
    - Loads application properties, SWB properties, and standard system configuration properties.
    - Validates the request to ensure all mandatory attributes are present.
    - Writes the request details to a JSON file with a timestamped filename.
    - Inserts the details into a queue for further processing.
    - Sets the response status to success if no exceptions are encountered.
    - Handles exceptions and updates the response object accordingly.

- Parameters:
    - structureSWBRequest

- Response:
    - Response will content status Success/Failure.

Note:

The below mentioned are some of few downstream names which are validated in the methods of some SOAP APIs:

- Digital Fulfillment
- Glovia
- SCDH(EBM)
- E2open (ITEM)
- PESO
- SPM (SPM)
- SDR
- Client Product Group (CPG)
- ESupport (ESP)
- Global Trade Management (GTM)