Veteran Affairs

**WSS 686 Documentation**

February 14, 2017

**Table Of Contents**

Table of Contents

[Purpose 5](#_Toc476225005)

[Audience 5](#_Toc476225006)

[History 5](#_Toc476225007)

[Architecture 5](#_Toc476225008)

[Code 6](#_Toc476225009)

[REST Services 6](#_Toc476225010)

[QueuedDocumentTransferExternalService 6](#_Toc476225011)

[SendMailMimeMessageService 7](#_Toc476225012)

[UserMessageServiceRestClient 8](#_Toc476225013)

[FindRatingInfoService 10](#_Toc476225014)

[JsonWebTokenCreationService 11](#_Toc476225015)

[SepCurrentUserService 12](#_Toc476225016)

[SepPoaAuthService 13](#_Toc476225017)

[UserProfileService 14](#_Toc476225018)

[Services 16](#_Toc476225019)

[AuditPersistanceService 16](#_Toc476225020)

[DocumentManagement 18](#_Toc476225021)

[InFlightFormService 18](#_Toc476225022)

[LoadForm686Service 21](#_Toc476225023)

[LookupService 22](#_Toc476225024)

[Form686ModelConversionService 23](#_Toc476225025)

[LoadSojHelper 23](#_Toc476225026)

[PDFManagementService 24](#_Toc476225027)

[PdfHeaderCreationService 27](#_Toc476225028)

[Form686Service 27](#_Toc476225029)

[Form674Service 28](#_Toc476225030)

[PdfCoverSheetCreationService 29](#_Toc476225031)

[SchoolLookUpService 30](#_Toc476225032)

[Form686SubmissionService 31](#_Toc476225033)

[SyncService 32](#_Toc476225034)

[TransactionLogUtilityService 33](#_Toc476225035)

[Batch Jobs 34](#_Toc476225036)

[Document 686 Job 34](#_Toc476225037)

[Submit 686 Job 40](#_Toc476225038)

[Post Submit 686 Job 43](#_Toc476225039)

[Use-Cases 63](#_Toc476225040)

[Top Level 63](#_Toc476225041)

[Add a New Dependent 64](#_Toc476225042)

[Add Education 65](#_Toc476225043)

[Add Marital Information 66](#_Toc476225044)

[Web Flow 67](#_Toc476225045)

[Dependent-flow 67](#_Toc476225046)

[Service Calls in “Dependent-flow” 67](#_Toc476225047)

[Marital-flow 67](#_Toc476225048)

[Divorce-flow 68](#_Toc476225049)

[Previous-flow(For prior marriages) 68](#_Toc476225050)

[Child-flow 68](#_Toc476225051)

[School-flow 69](#_Toc476225052)

[Upload-flow 70](#_Toc476225053)

[Appendix 70](#_Toc476225054)

[BGS Service WSDLs 70](#_Toc476225055)

[AwardWebService 70](#_Toc476225056)

[BenefitClaimService 70](#_Toc476225057)

[ClaimantWebService 70](#_Toc476225058)

[RORoutingService 70](#_Toc476225059)

[ShareStandardDataWebService 70](#_Toc476225060)

[StandardDataWebService 71](#_Toc476225061)

[VeteranRecordService 71](#_Toc476225062)

[VRMDraftFormDocumentService 71](#_Toc476225063)

[SOAP-UI Tests 71](#_Toc476225064)

[AwardDataService 71](#_Toc476225065)

[BenefitClaimService 71](#_Toc476225066)

[VetRecordWebService 71](#_Toc476225067)

[VETSNET 71](#_Toc476225068)

[VRMDraftFormDocumentService 71](#_Toc476225069)

[System Design Document 71](#_Toc476225070)

[ICD Document 72](#_Toc476225071)

[Wireframes 72](#_Toc476225072)

[Test Cases 72](#_Toc476225073)

[Glossary 72](#_Toc476225074)

# Purpose

The purpose of this document is to capture 686 as it exists today. This document will breakdown the code in 686 Services and 686 Webparts by Architecture, REST Services, Services, Batch Jobs, and Web Flows. The document should business systems analysts, architects, and developers the information they need to have an understanding of the code structure and flow. Any WSDLs, WADLs, SOAP-UI Tests will be linked in the Appendix. Any terms or acronyms not likely to be understood will be covered in the Glossary.

**\* Code marked in red is code that was discovered during this documentation that needs to be refactored due to violation of common architectural design principles.**

# Audience

This document is intended technical resources; business system analysts, architects, and developers. This document is not intended for general business readers or individuals with little or no technical background.

# History

|  |  |  |
| --- | --- | --- |
| Date | Description | Author |
| 02/14/2017 | Initial Draft | David Cano |
| 3/2/2017 | Created Word document 1.0 | Sophia Nawab |
|  |  |  |

# Architecture

The 686 services follow design principles common to service oriented architectures. The most prevalent pattern the technical resource will see throughout the code is the use of the Request, Response, and Service pattern. Other patterns the technical resource should see are encapsulation, open/close principle, abstraction, and separation of concerns. Some services in 686 are pass through to claims, common, and document services. And some services are calls to partner services, external the EVSS projects. Where possible, these WSDLs, WADLs, and SOAP-UI Tests will also be included in the documentation or Appendix.

# Code

The following documents the code as it exists today. The level of detail is intended to give technical resources a clear understanding of the interfaces and contracts that exist, and a summary of the logic executed.

## REST Services

The following are the REST services that exist in the wss-686-services project. Most of these REST services are passed through to Common or Document Services. They service as a layer of abstraction and protection to the 686 code base. The 686 code base knows only these REST service calls, and not the services that exist in Common or Document Services.

### QueuedDocumentTransferExternalService

This service is a pass through to Document Services. The purpose of this service is call out to Document Services to add an uploaded document to the queue.

**QueuedDocumentTransferExternalServiceClient**

Method:addStagedDocumentToQueue

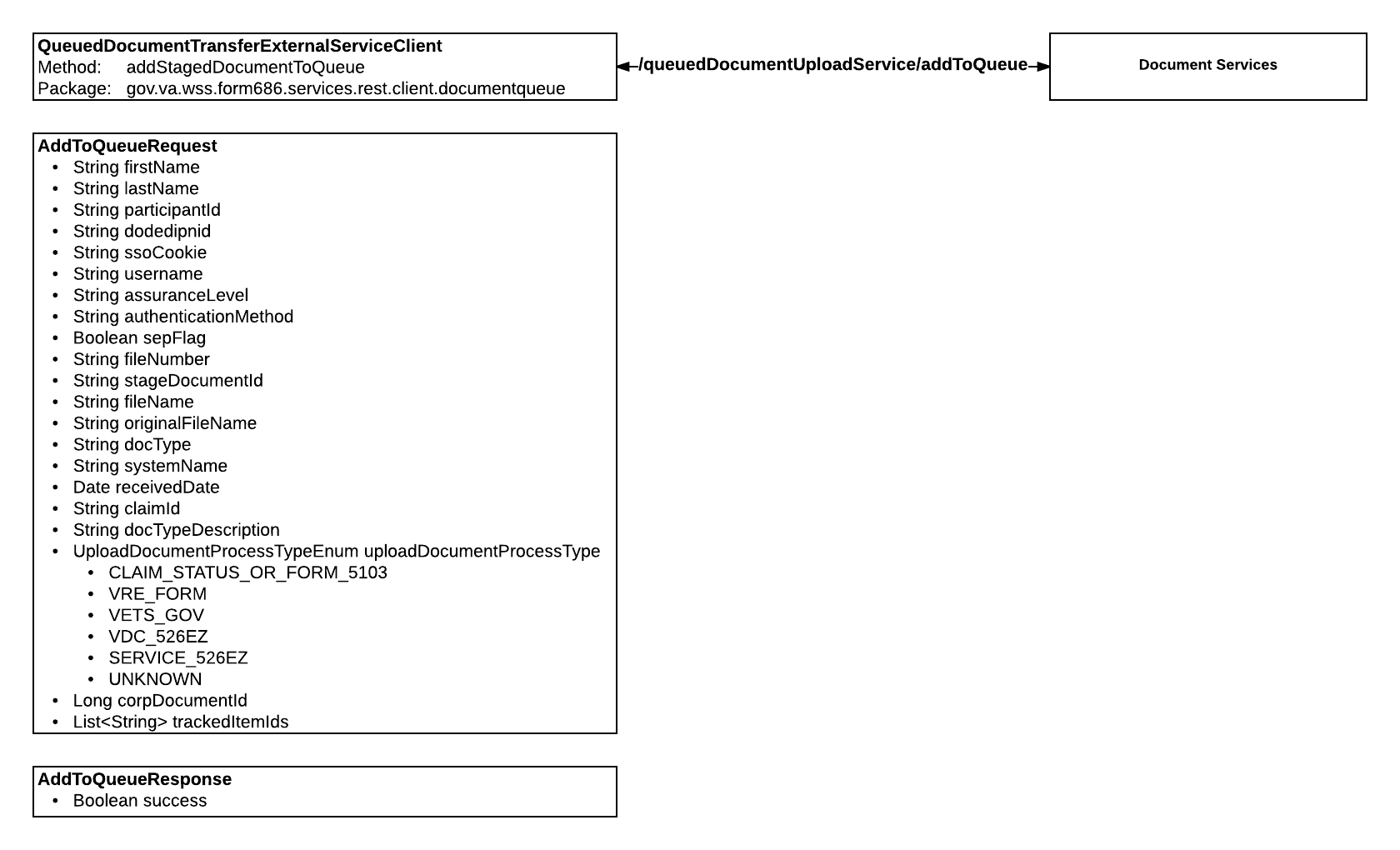
Package: gov.va.wss.form686.services.rest.client.documentqueue

**AddToQueueRequest**

* String firstName
* String lastName
* String participantId
* String dodedipnid
* String ssoCookie
* String username
* String assuranceLevel
* String authenticationMethod
* Boolean sepFlag
* String fileNumber
* String stageDocumentId
* String fileName
* String originalFileName
* String docType
* String systemName
* Date receivedDate
* String claimId
* String docTypeDescription
* UploadDocumentProcessTypeEnum uploadDocumentProcessType
  + CLAIM\_STATUS\_OR\_FORM\_5103
  + VRE\_FORM
  + VETS\_GOV
  + VDC\_526EZ
  + SERVICE\_526EZ
  + UNKNOWN
* Long corpDocumentId
* List<String> trackedItemIds

**AddToQueueResponse**

* Boolean success



### SendMailMimeMessageService

This service is a pass through to Common Services. The purpose of this service is to send an email message to Common Services to be sent out.

**SendMailMimeMessageServiceRestClient**

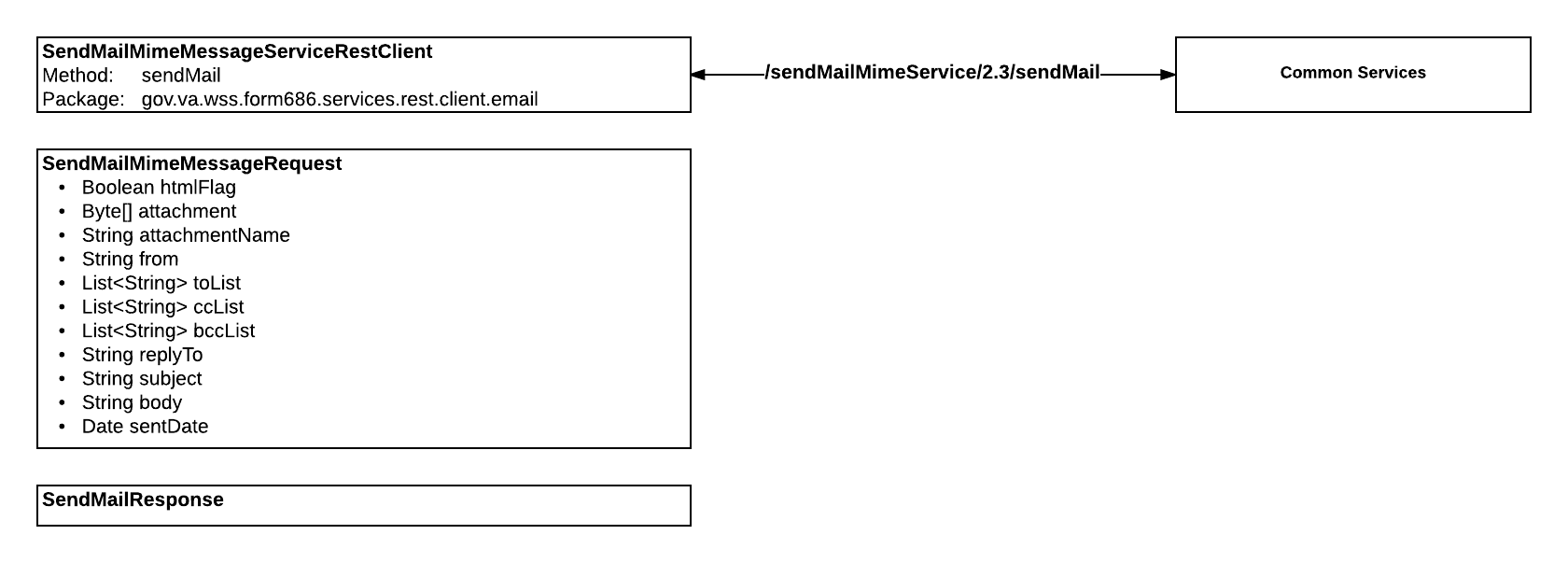
Method:sendMail

Package: gov.va.wss.form686.services.rest.client.email

**SendMailMimeMessageRequest**

* Boolean htmlFlag
* Byte[] attachment
* String attachmentName
* String from
* List<String> toList
* List<String> ccList
* List<String> bccList
* String replyTo
* String subject
* String body
* Date sentDate

**SendMailResponse**



### UserMessageServiceRestClient

This services is a pass through to Common Services. The purpose of this service is to send User Messages to Common Services to be added to the Message Center.

**UserMessageServiceRestClient**

Method:sendUserMessages

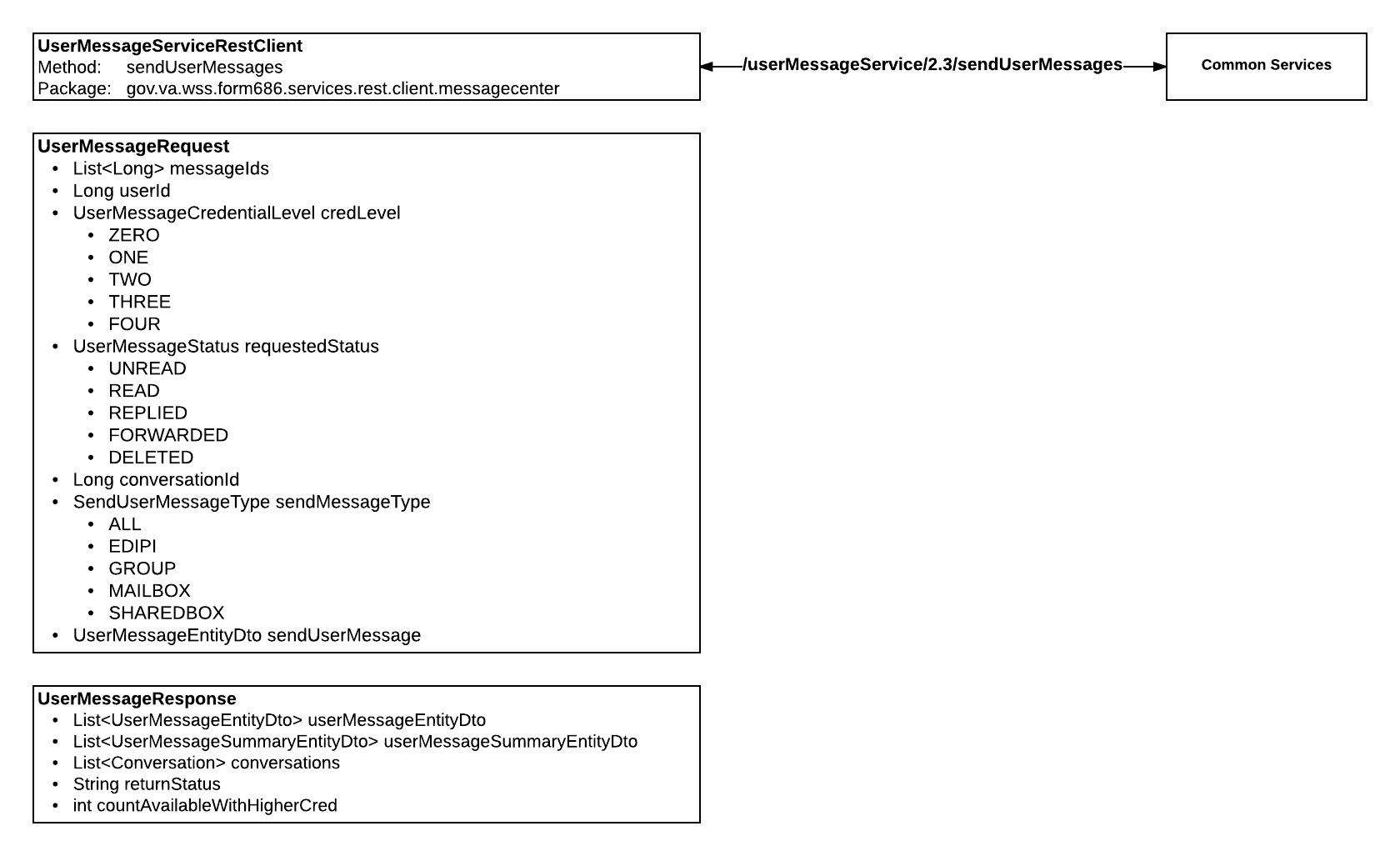
Package: gov.va.wss.form686.services.rest.client.messagecenter

**UserMessageRequest**

* List<Long> messageIds
* Long userId UserMessage
* CredentialLevel credLevel
  + ZERO
  + ONE
  + TWO
  + THREE
  + FOUR
* UserMessageStatus requestedStatus
  + UNREAD
  + READ
  + REPLIED
  + FORWARDED
  + DELETED
* Long conversationId
* SendUserMessageType sendMessageType
  + ALL
  + EDIPI
  + GROUP
  + MAILBOX
  + SHAREDBOX
* UserMessageEntityDto sendUserMessage

**UserMessageResponse**

* List<UserMessageEntityDto> userMessageEntityDto
* List<UserMessageSummaryEntityDto> userMessageSummaryEntityDto
* List<Conversation> conversations
* String returnStatus
* int countAvailableWithHigherCred



### FindRatingInfoService

This service is a pass through to Common Services. The purpose of this service is to use the Participant ID to find Rating Information for the Veteran, their spouse, or their dependents.

**FindRatingInfoServiceRestClient**

Method:findRatingDataByParticipantId

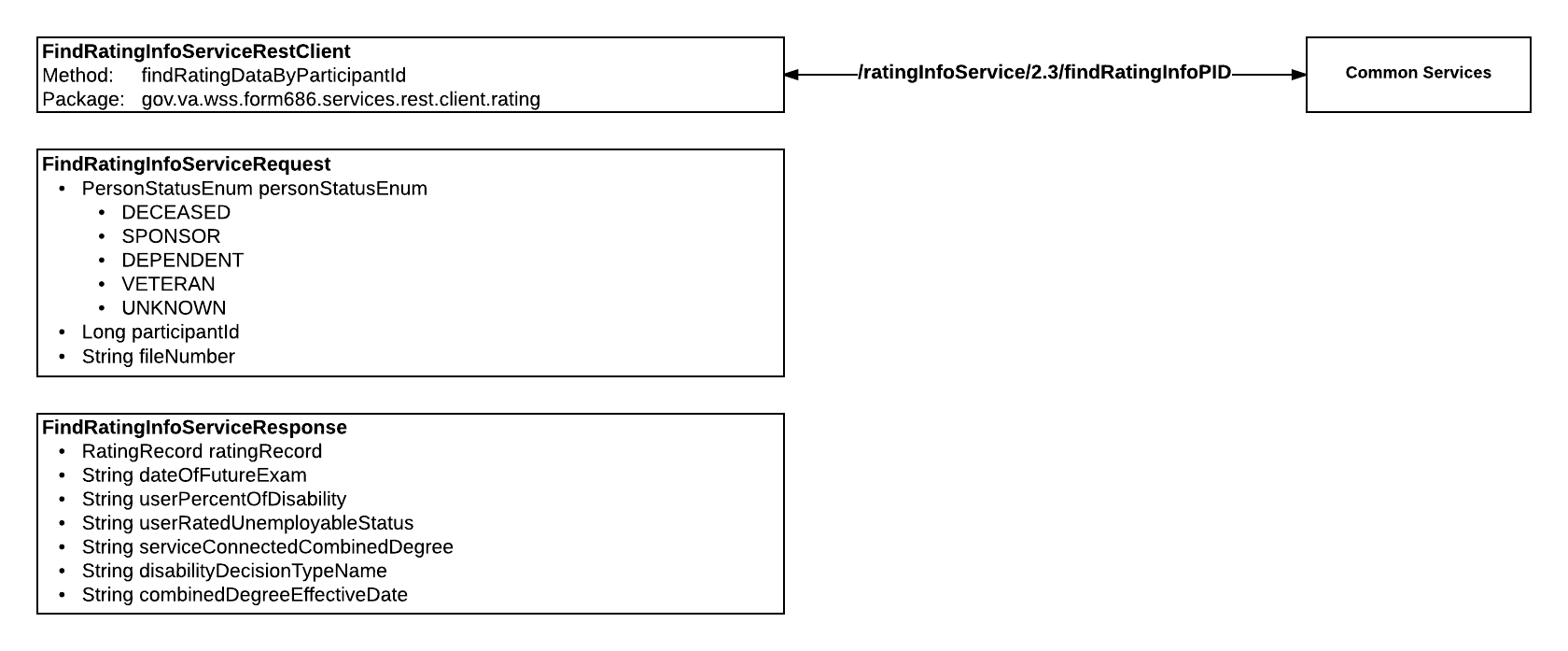
Package: gov.va.wss.form686.services.rest.client.rating

**FindRatingInfoServiceRequest**

* PersonStatusEnum personStatusEnum
  + DECEASED
  + SPONSOR
  + DEPENDENT
  + VETERAN
  + UNKNOWN
* Long participantId
* String fileNumber

**FindRatingInfoServiceResponse**

* RatingRecord ratingRecord
* String dateOfFutureExam
* String userPercentOfDisability
* String userRatedUnemployableStatus
* String serviceConnectedCombinedDegree
* String disabilityDecisionTypeName
* String combinedDegreeEffectiveDate



### JsonWebTokenCreationService

This service is a pass through to Common Services. The purpose of this service is to create a JSON web token to be used by the presentation tier.

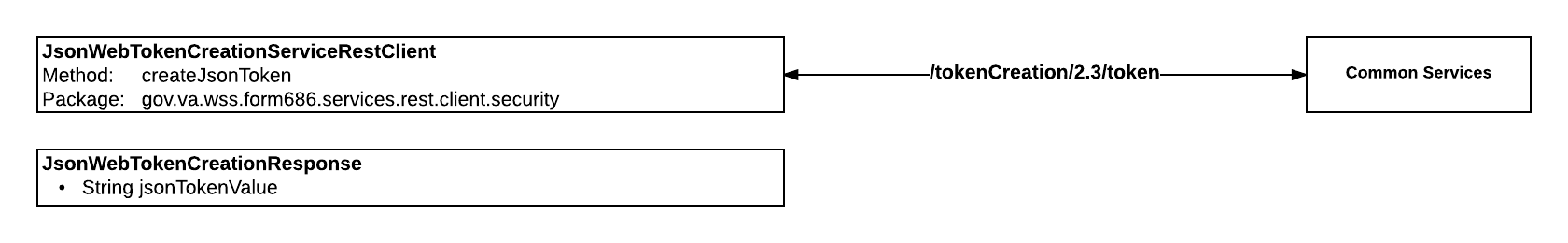
**JsonWebTokenCreationServiceRestClient**

Method:createJsonToken

Package: gov.va.wss.form686.services.rest.client.security

**JsonWebTokenCreationResponse**

* String jsonTokenValue



### SepCurrentUserService

This service is a pass through to Common Services. The purpose of this service is to retrieve the Current User from SEP.

**SepCurrentUserServiceRestClient**

Method:getCurrentUserInformation

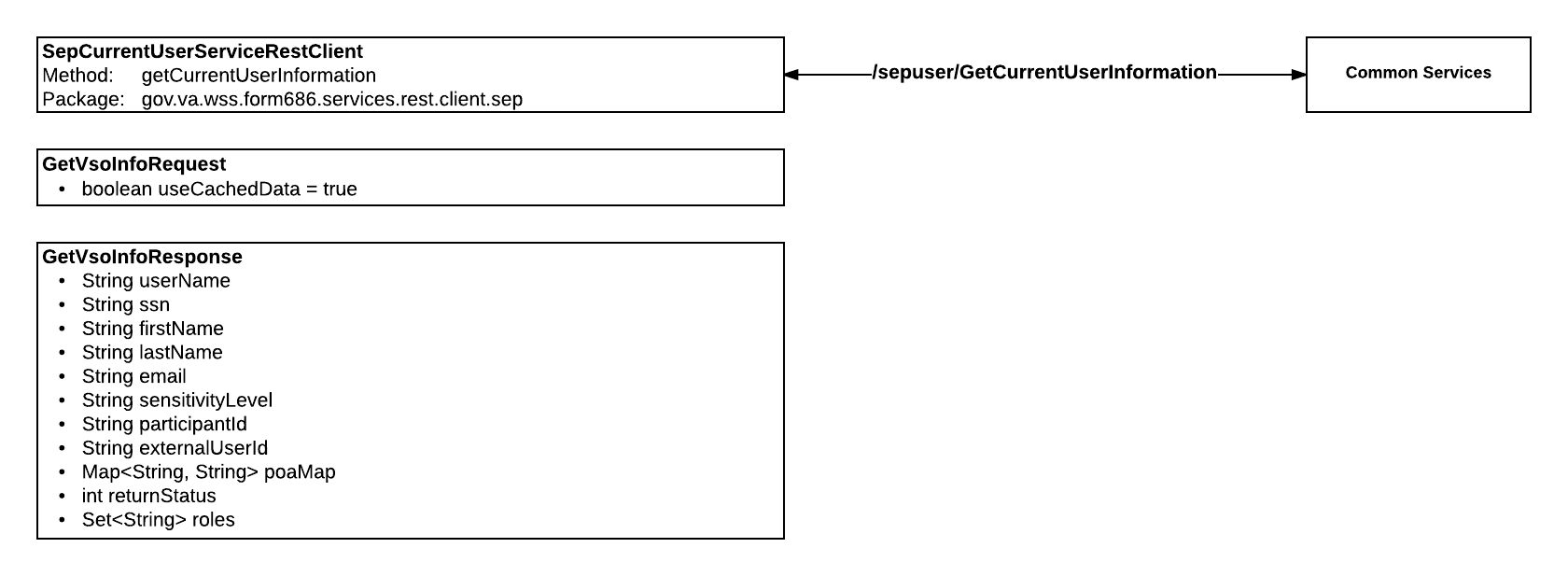
Package: gov.va.wss.form686.services.rest.client.sep

**GetVsoInfoRequest**

* boolean useCachedData = true

**GetVsoInfoResponse**

* String userName
* String ssn
* String firstName
* String lastName
* String email
* String sensitivityLevel
* String participantId
* String externalUserId
* Map<String, String> poaMap
* int returnStatus
* Set<String> roles



### SepPoaAuthService

This service is a pass through to Common Services. The purpose of this service is to retrieve the current SEP POA Authorization for the User. The second purpose of this service is to try to create the SEP POA Authorization for the User, if it does not exist.

**SepPoaAuthServiceRestClient**

Method:getCurrentPoaAuth

Package: gov.va.wss.form686.services.rest.client.sep

**GetCurrentPoaAuthResponse**

* Agent agent
* Principal principal
* int returnStatus



**SepPoaAuthServiceRestClient**

Method:createPoaAuth

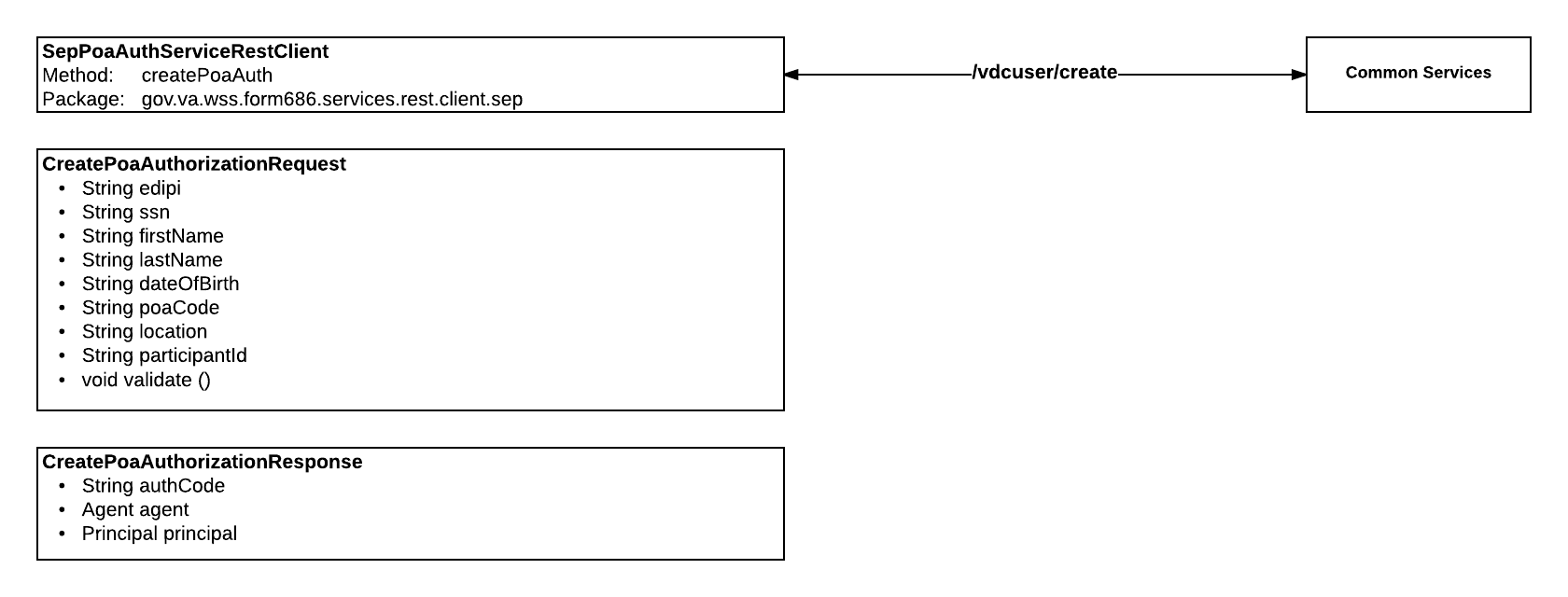
Package: gov.va.wss.form686.services.rest.client.sep

**CreatePoaAuthorizationRequest**

* String edipi
* String ssn
* String firstName
* String lastName
* String dateOfBirth
* String poaCode
* String location
* String participantId
* **void validate ()** - Validation should be done by the validation layer. This is a violation of the separation of concerns.

**CreatePoaAuthorizationResponse**

* String authCode
* Agent agent
* Principal principal



### UserProfileService

This service is a pass through to Common Services. The purpose of this service is the following:

* Retrieve the User Icon
* Retrieve the Participant ID
* Retrieve the BIRLS File Number
* Retrieve the Beneficiary Profile
* Retrieve the Beneficiary Profile by Participant ID
* Retrieve the Representative Profile

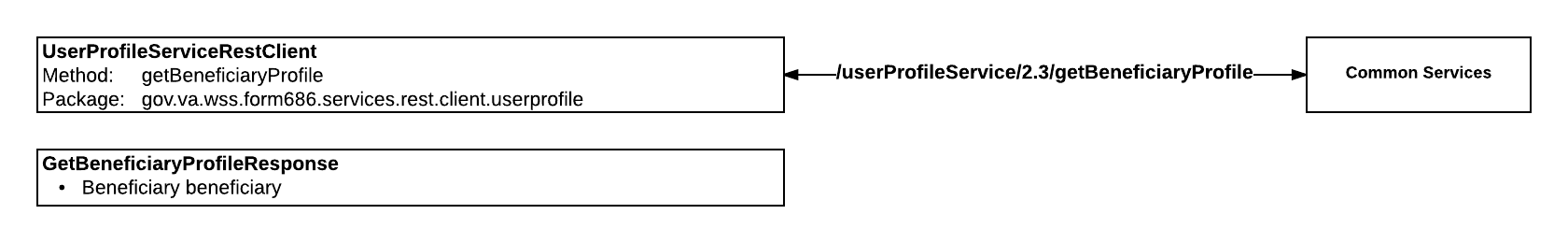
**UserProfileServiceRestClient**

Method:getBeneficiaryProfile

Package: gov.va.wss.form686.services.rest.client.userprofile

**GetBeneficiaryProfileResponse**

* Beneficiary beneficiary



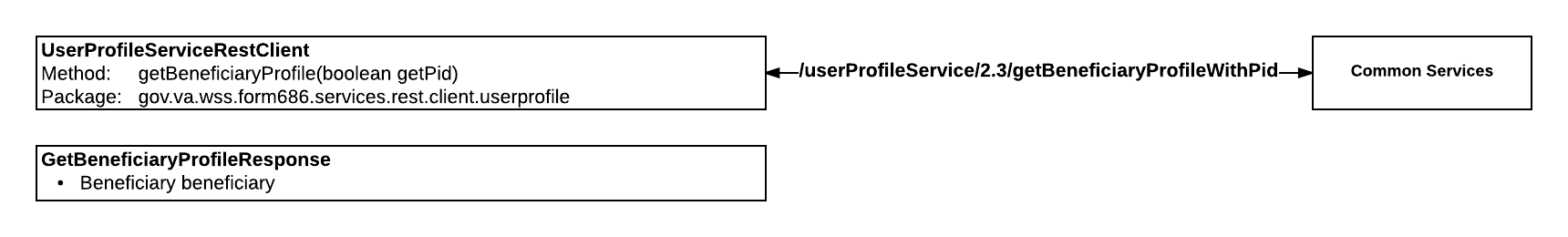
**UserProfileServiceRestClient**

Method:getBeneficiaryProfile(boolean getPid)

Package: gov.va.wss.form686.services.rest.client.userprofile

**GetBeneficiaryProfileResponse**

* Beneficiary beneficiary



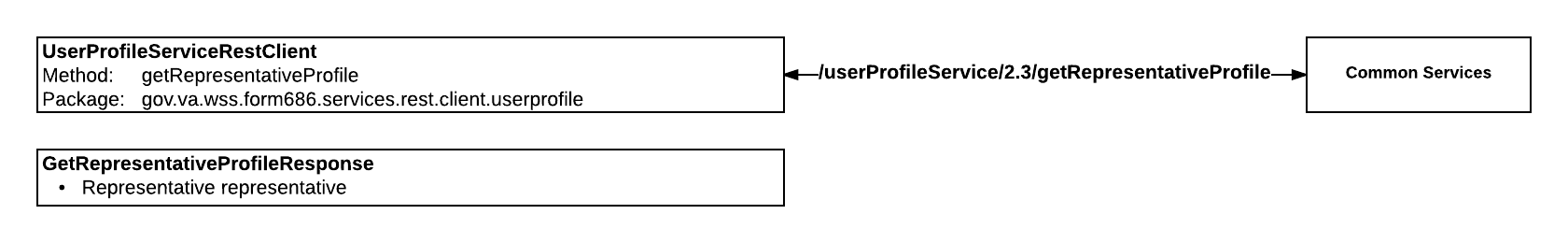
**UserProfileServiceRestClient**

Method:getRepresentativeProfile

Package: gov.va.wss.form686.services.rest.client.userprofile

**GetRepresentativeProfileResponse**

* Representative representative



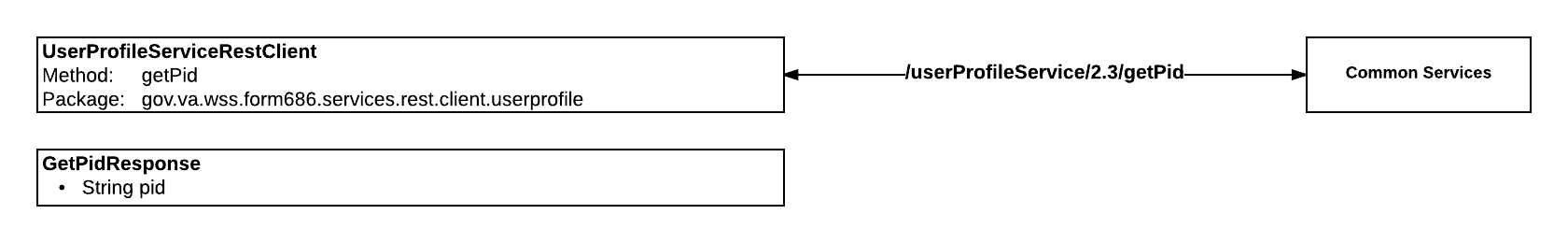
**UserProfileServiceRestClient**

Method:getPid

Package: gov.va.wss.form686.services.rest.client.userprofile

**GetPidResponse**

* String pid



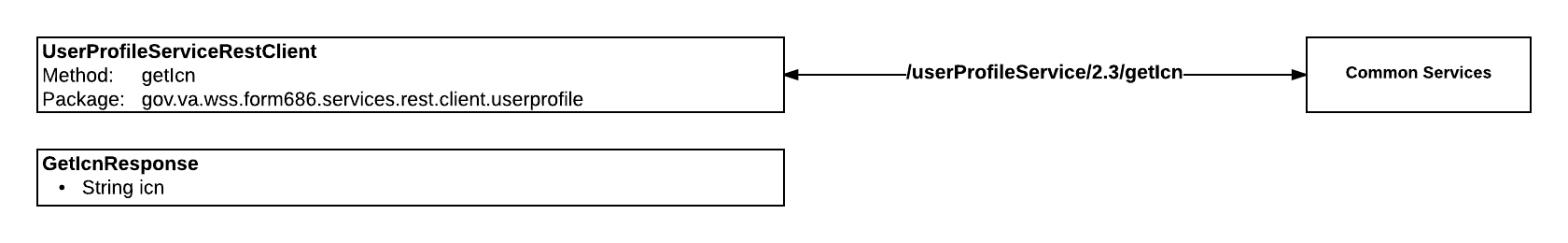
**UserProfileServiceRestClient**

Method:getIcn

Package: gov.va.wss.form686.services.rest.client.userprofile

**GetIcnResponse**

* String icn



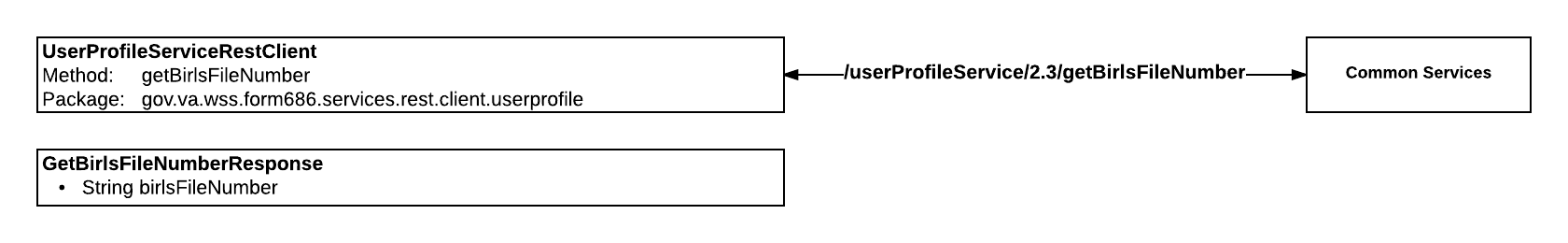
**UserProfileServiceRestClient**

Method:getBirlsFileNumber

Package: gov.va.wss.form686.services.rest.client.userprofile

**GetBirlsFileNumberResponse**

* String birlsFileNumber



## Services

These are the Services defined in wss-686-services project. These services can be called directly by other 686 services or by 686 web controllers. These are the services that do the bulk of the heavy lifting in the wss-686-services project.

### AuditPersistanceService

This service does the following:

* Saves a Process Status and a Process Comment
* Updates the Last Updated Date on the User Process Mapper

**AuditPersistanceServiceImpl**

Method: auditStatusChange

Package: gov.va.wss.form686.services.service.audit

**AuditPersistanceRequest**

String edipi

String userId

String poaCode

String lastName

String middleName

String firstName

String userRole

boolean reviewIndicator

String userCorpId

String statusCode

String procId

String inflightId

String comment

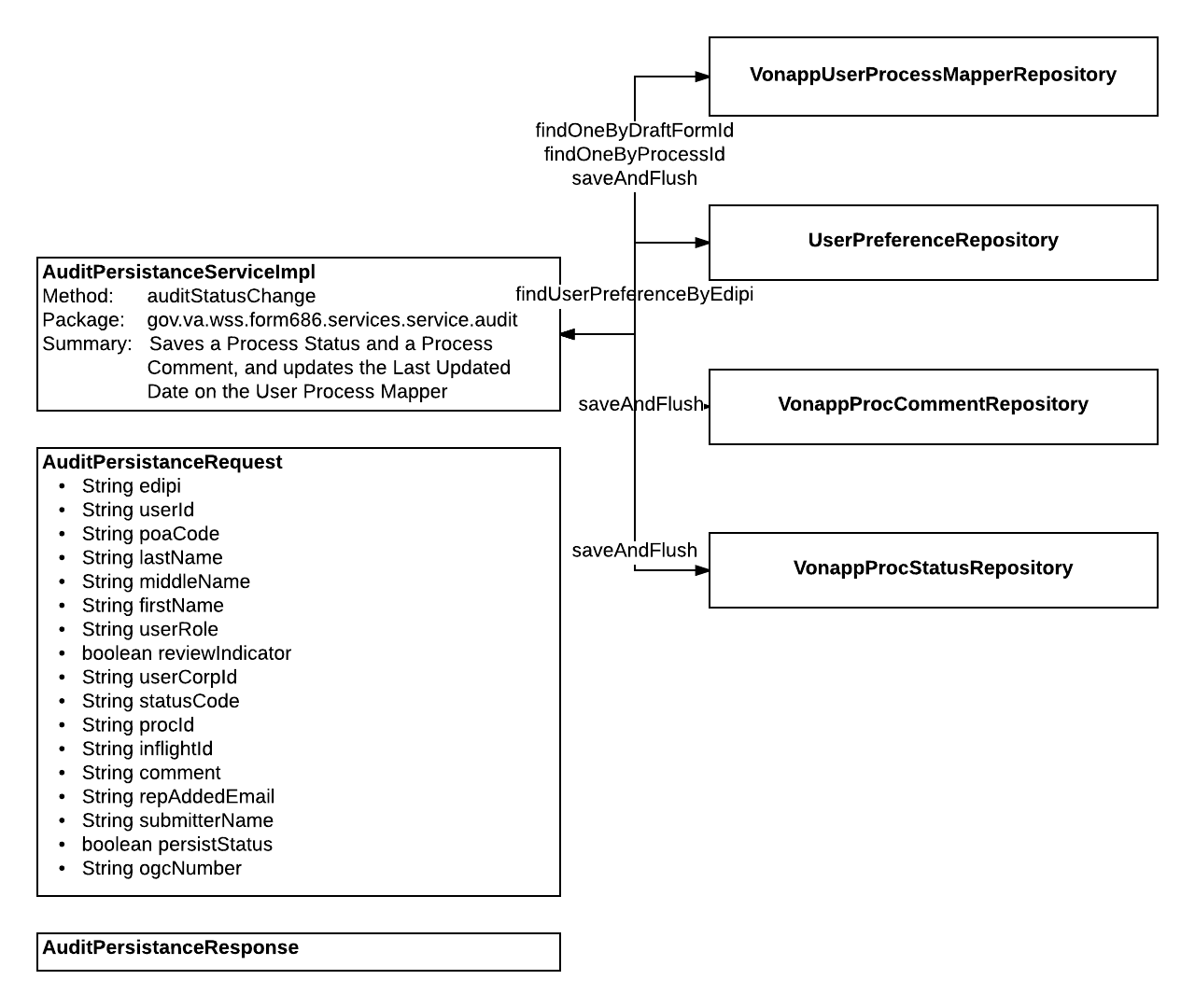
String repAddedEmail

String submitterName

boolean persistStatus

String ogcNumber

**AuditPersistanceResponse**



### DocumentManagement

This service is the Document Management service for 686 that retrieves Submitted Documents. **But it actually calls the VDC Shared Services to let VDC Shared Services handle document management. There is a call in Document Services that does the same thing.** It seems like the original document management code was written in VDC Shared Services, and rather than encapsulating that code in it’s own dependency library that other projects could consume, the developers instead decided to create circular dependencies between VDC, 686, and Document Services with these calls. The other issue with this service is its concrete implementation class exists in wss-686-webparts instead of wss-686-services where it belongs. My guess as to why the concrete class was put here is likely because the circular dependencies of putting it in wss-686-services probably made it near impossible for developers to get a clean build of their projects. I’m also not certain why this service sends back a SendMailResponse versus a Response object that makes sense and also returns a List of Submitted Documents.

**DocumentManagementRestClient**

**Method: submittedDocuments**

**Package: gov.va.wss.form686.webparts.rest.client.document**

**Extends: DocumentManagement**

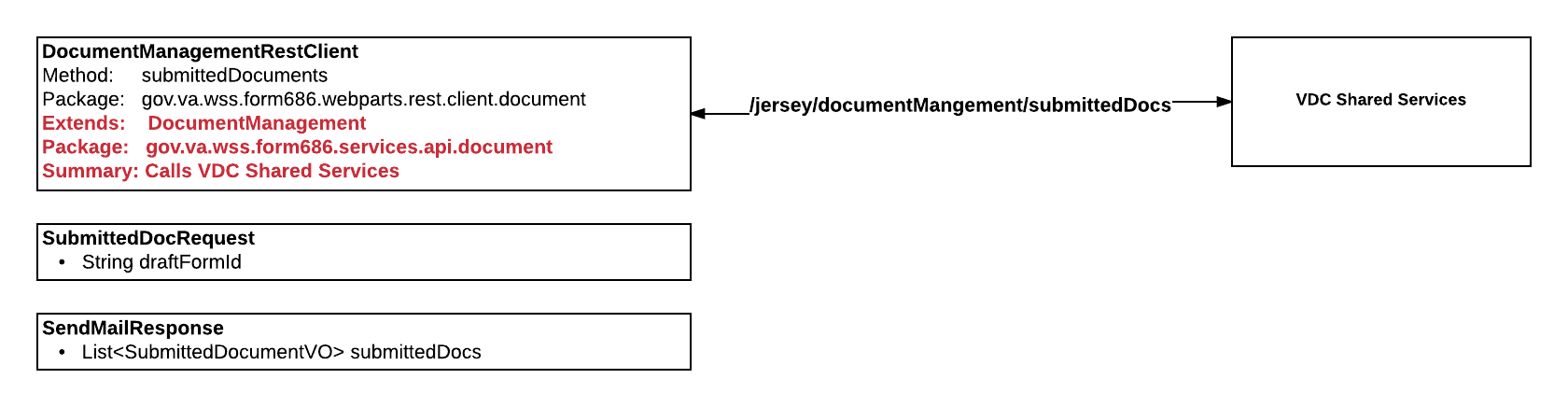
**Package: gov.va.wss.form686.services.api.document**

**SubmittedDocRequest**

String draftFormId

**SendMailResponse**

List<SubmittedDocumentVO> submittedDocs



### InFlightFormService

This service does the following:

* Saves Form to VRM
  + Saves User Process Mapper to   InFlight Data Store
* Validates the SubmitProcess.Veteran object
* Cleans up the SubmitProcess object

**InFlightFormServiceImpl**

Method:saveForm

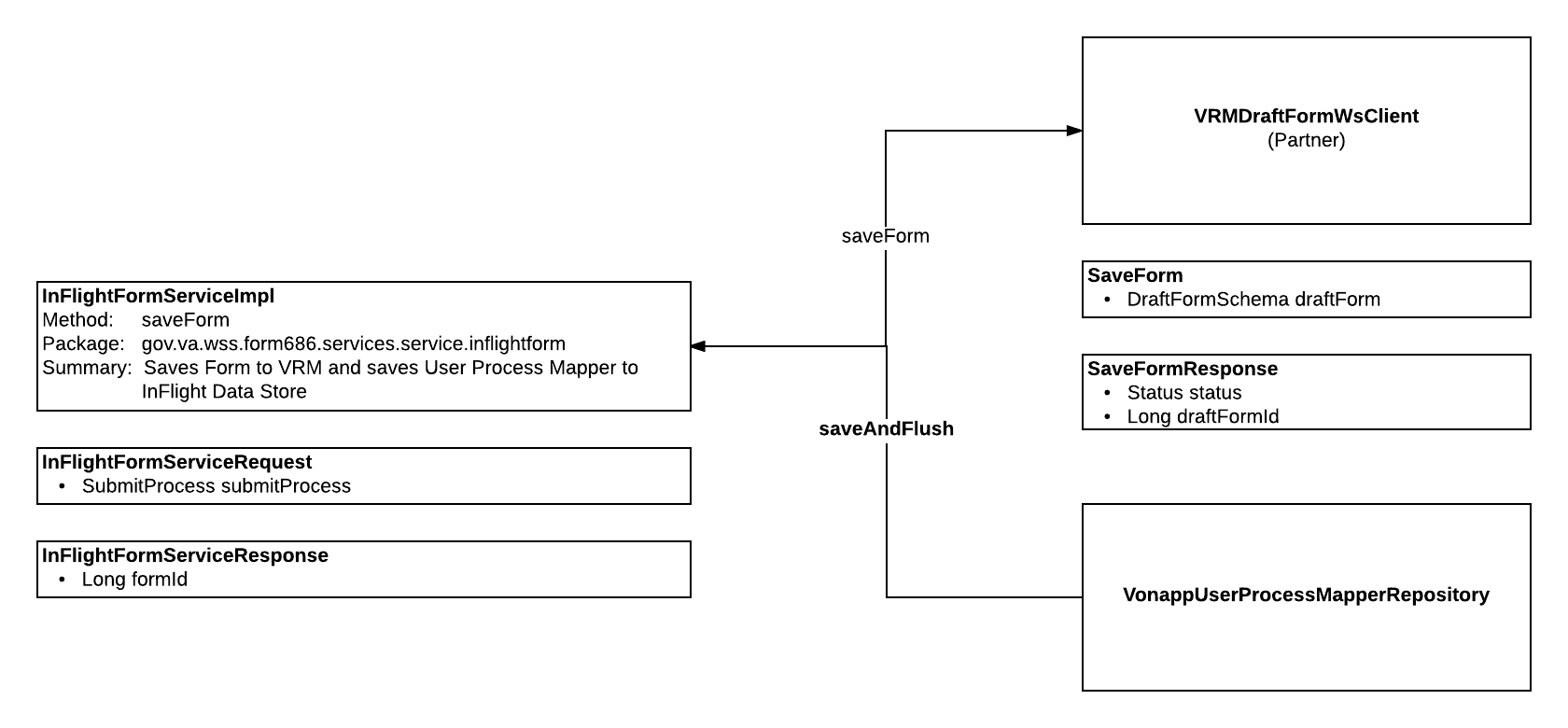
Package: gov.va.wss.form686.services.service.inflightform

**InFlightFormServiceRequest**

* SubmitProcess submitProcess

**InFlightFormServiceResponse**

* Long formId



**InFlightFormServiceImpl**

Method:validateForm

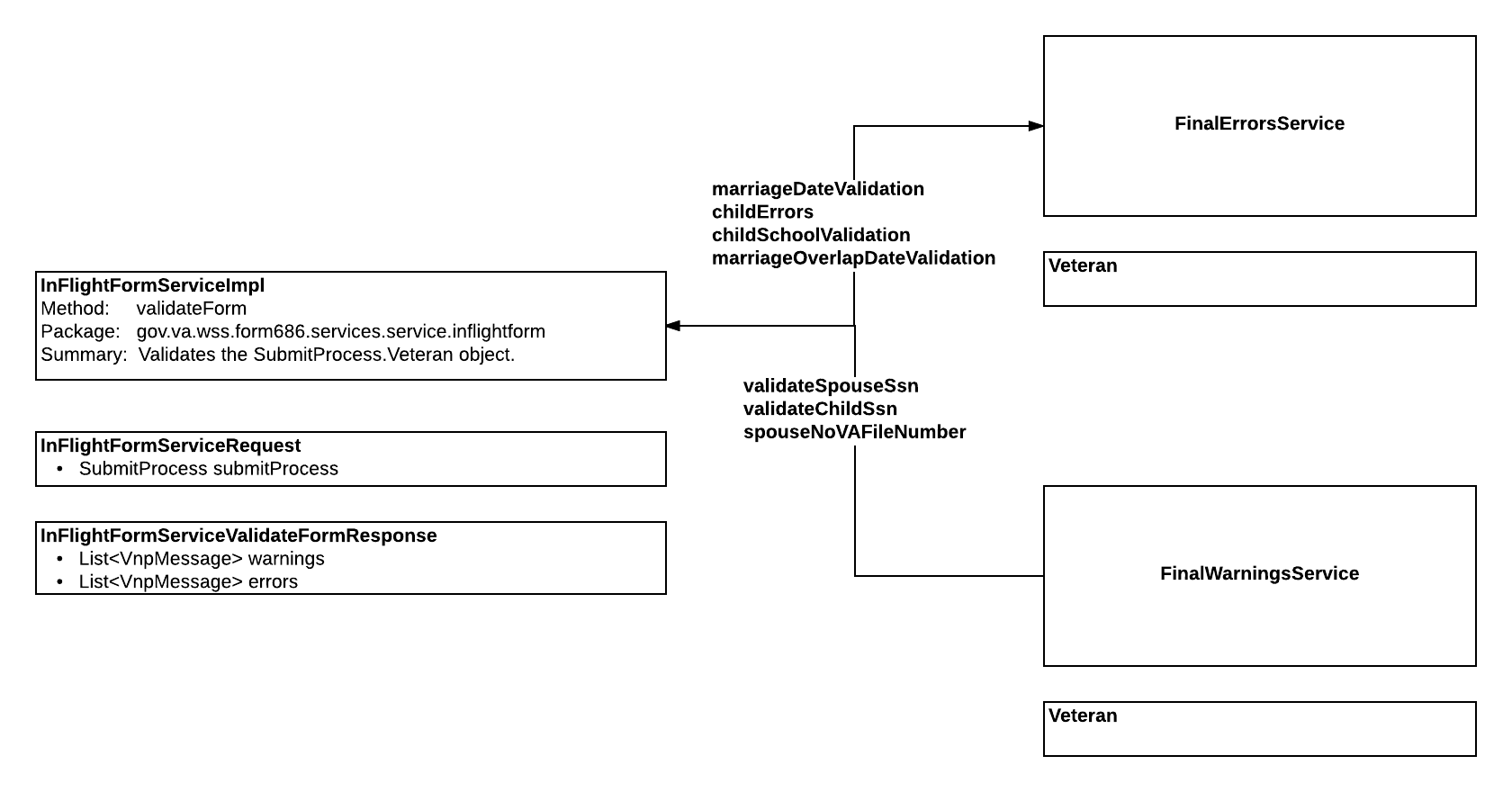
Package: gov.va.wss.form686.services.service.inflightform

**InFlightFormServiceRequest**

* SubmitProcess submitProcess

**InFlightFormServiceValidateFormResponse**

* List<VnpMessage> warnings
* List<VnpMessage> errors



**InFlightFormServiceImpl**

Method:cleanForm

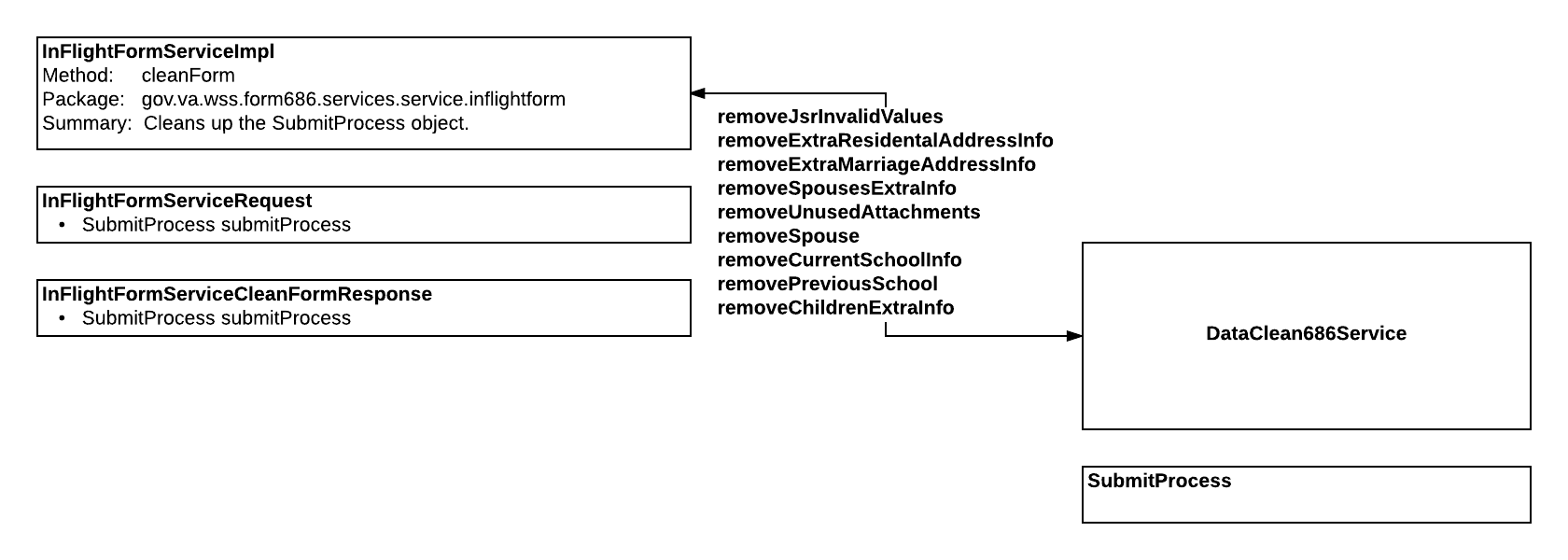
Package: gov.va.wss.form686.services.service.inflightform

**InFlightFormServiceRequest**

* SubmitProcess submitProcess

**InFlightFormServiceCleanFormResponse**

* SubmitProcess submitProcess



### LoadForm686Service

This service does the following:

* Find Current POA Auth
* Finds all open claims
* Finds VRM Draft Form and converts it
* Finds the Veteran record
* Finds the Ranting Information
* Find Station of Jurisdiction
* Then logs a User Process Mapper record
* Gets VRM Draft Form and converts it

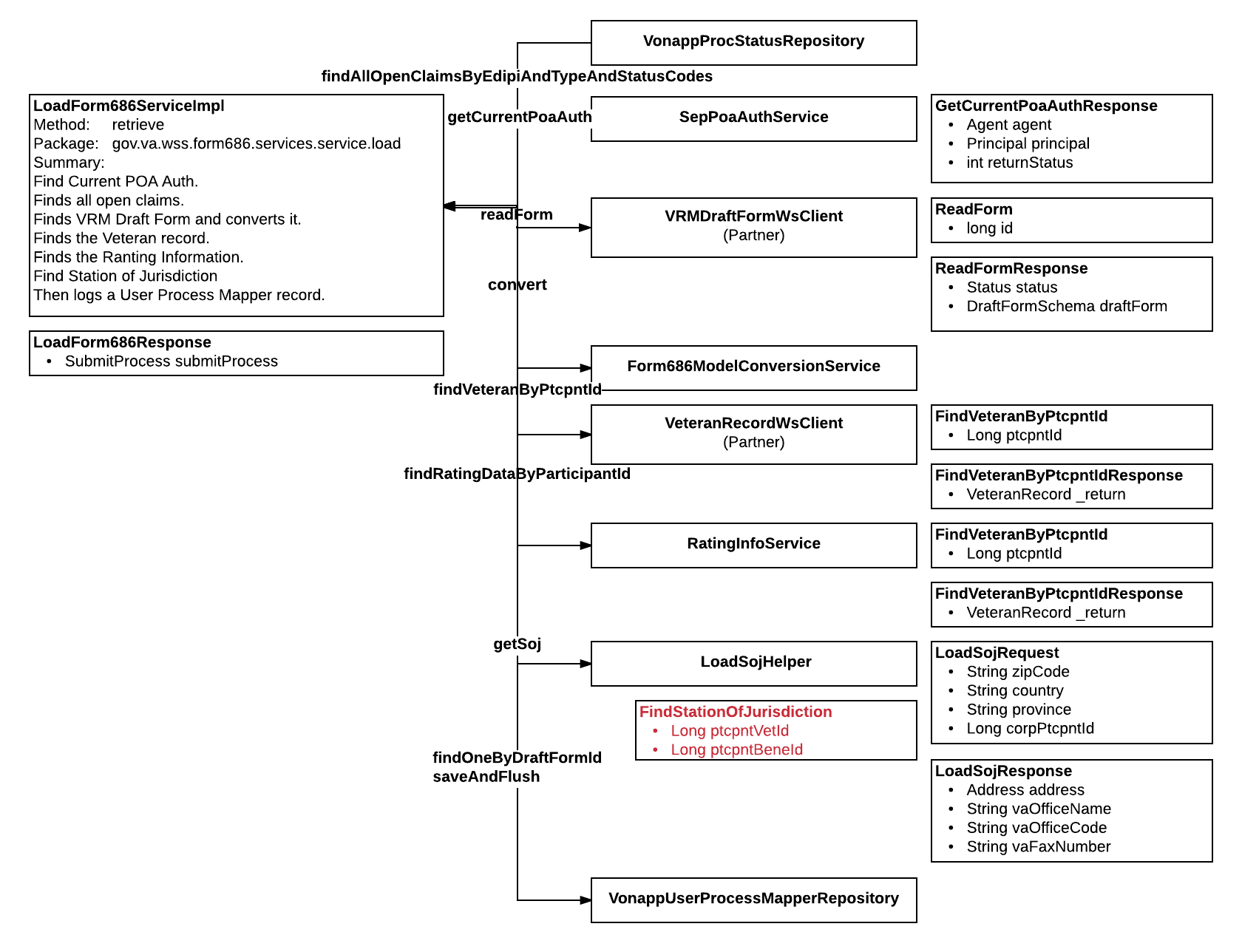
**LoadForm686ServiceImpl**

Method:retrieve

Package: gov.va.wss.form686.services.service.load

**LoadForm686Response**

* SubmitProcess submitProcess



**LoadForm686ServiceImpl**

Method:loadProcess

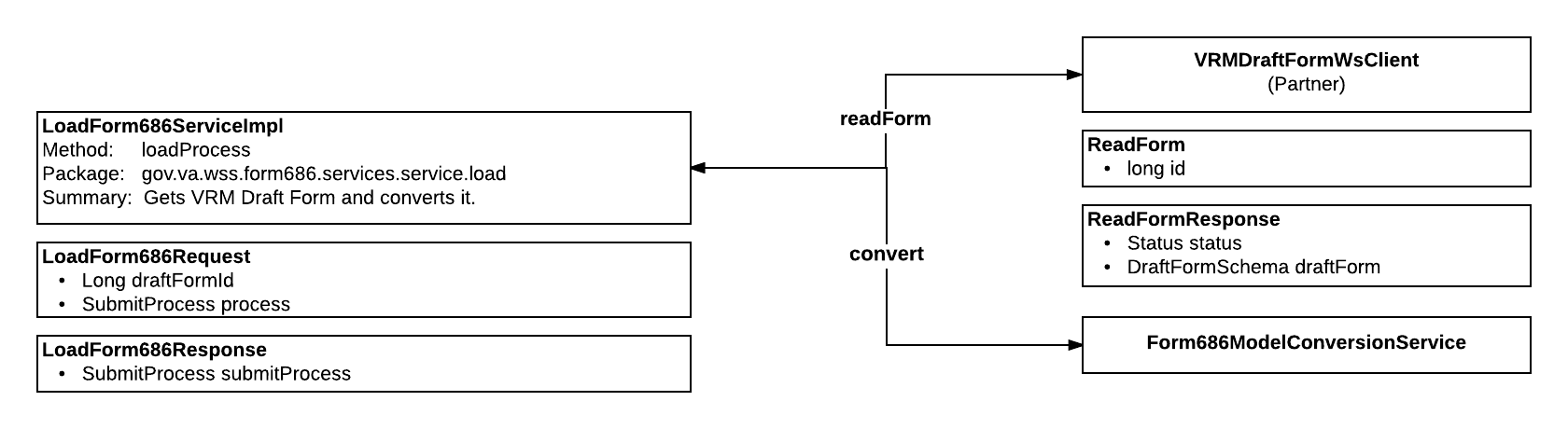
Package: gov.va.wss.form686.services.service.load

**LoadForm686Request**

* Long draftFormId
* SubmitProcess process

**LoadForm686Response**

* SubmitProcess submitProcess



### LookupService

This service does the following:

* Finds States and converts to a Map
* Finds Countries and converts to a Map

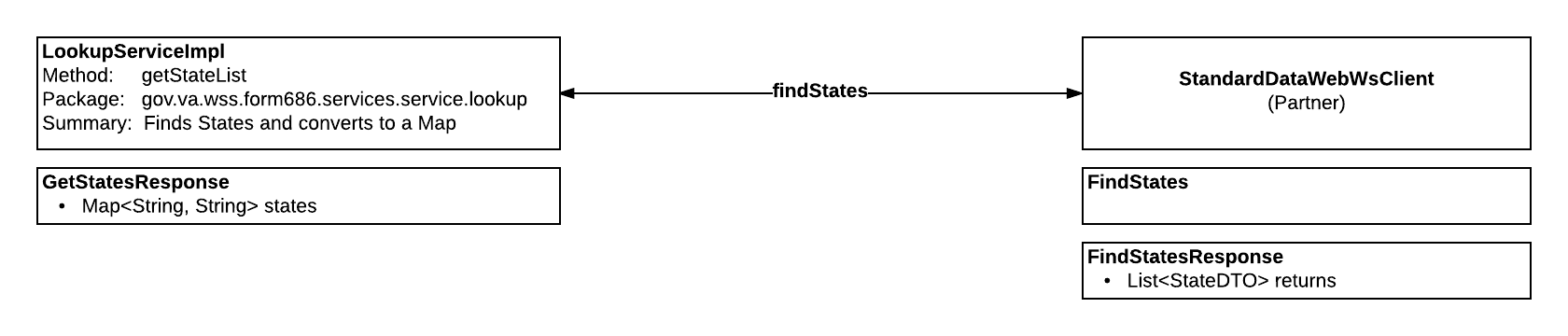
**LookupServiceImpl**

Method:getStateList

Package: gov.va.wss.form686.services.service.lookup

**GetStatesResponse**

* Map<String, String> states



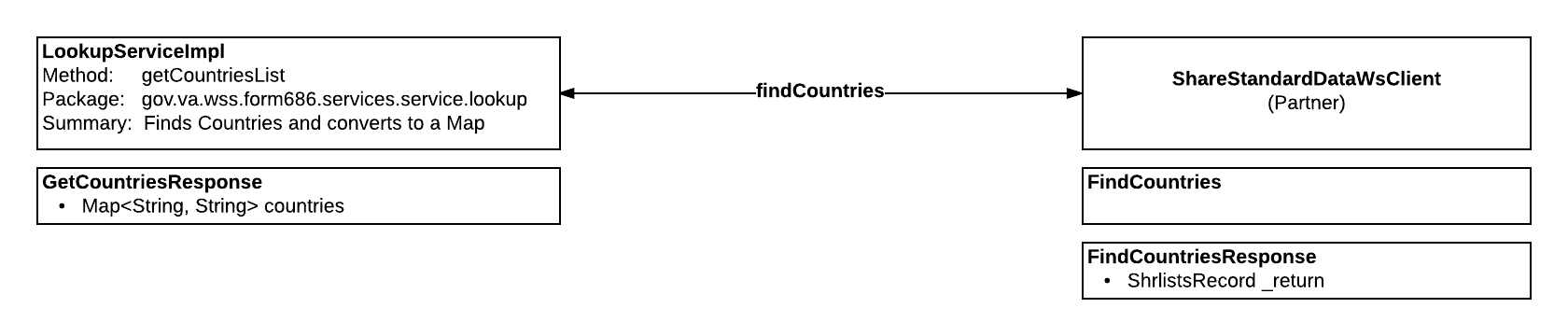
**LookupServiceImpl**

Method:getCountriesList

Package: gov.va.wss.form686.services.service.lookup

**GetCountriesResponse**

* Map<String, String> countries



### Form686ModelConversionService

The purpose of this service is to convert serialized form data.

**Form686ModelConversionServiceImpl**

Method:convert

Package: gov.va.wss.form686.services.service.load.converter

**Object**



### LoadSojHelper

The purpose of this service is to find the Station of Jurisdiction.

**LoadSojHelper**

Method:getSoj

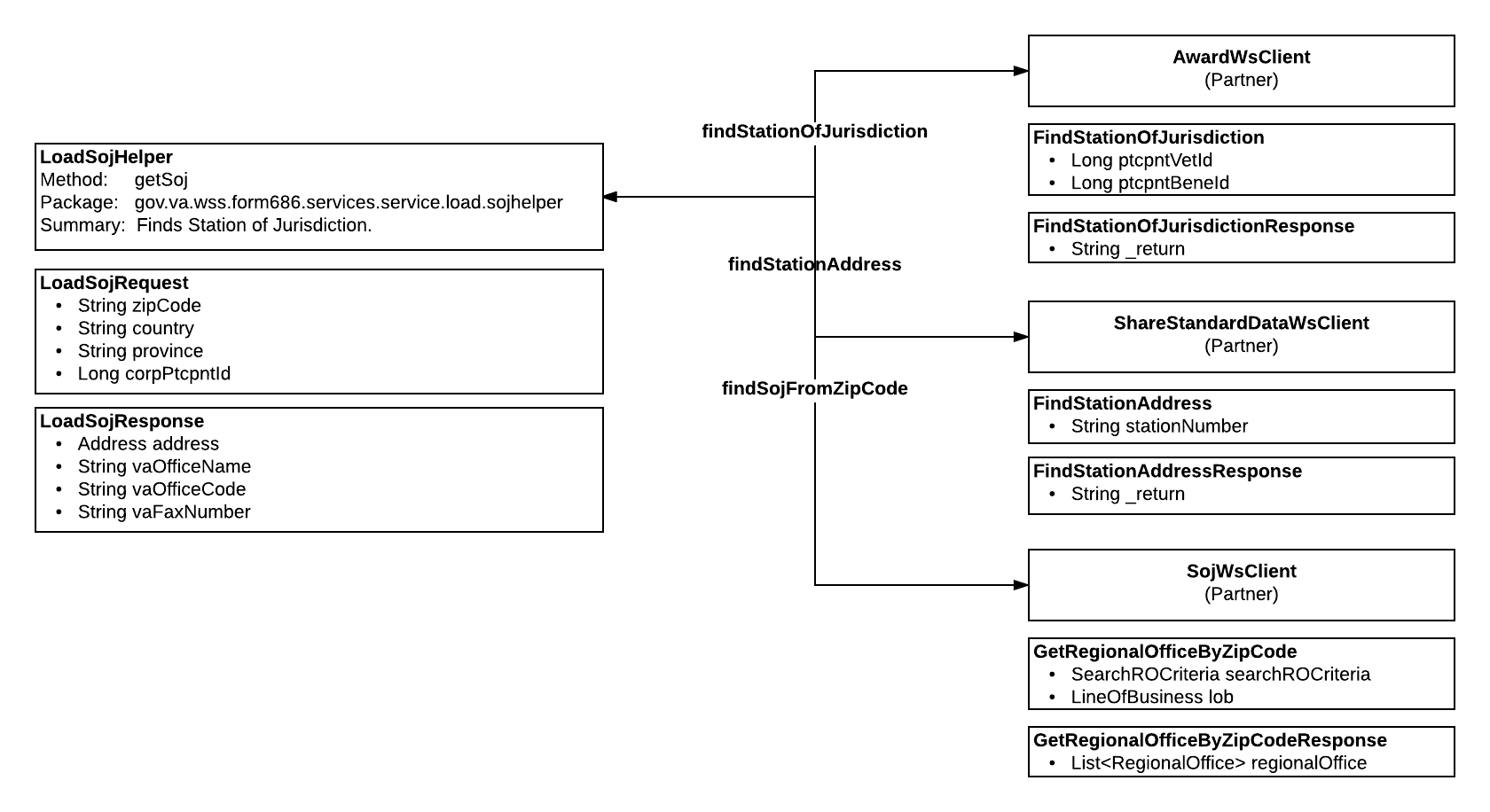
Package: gov.va.wss.form686.services.service.load.sojhelper

**LoadSojRequest**

* String zipCode
* String country
* String province
* Long corpPtcpntId

**LoadSojResponse**

* Address address
* String vaOfficeName
* String vaOfficeCode
* String vaFaxNumber



### PDFManagementService

This service is about generating PDFs and does the following:

* Generates the PDF Header
* Generates the 686 PDF
* Generates the 674 PDF
* Generates the PDF Cover Sheet

**PDFManagementServiceImpl**

Method:addHeaderToPDF

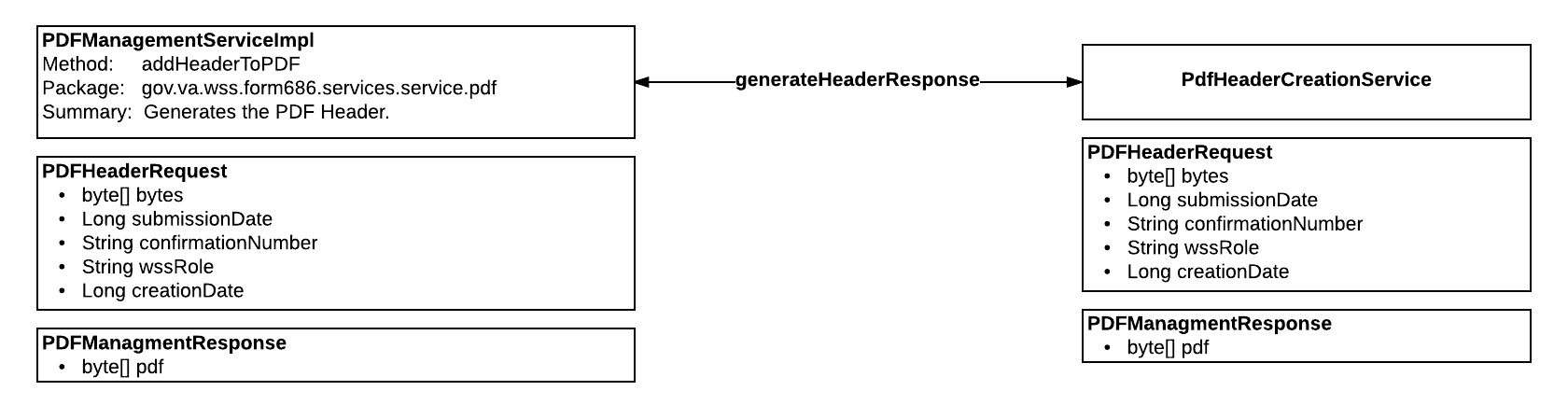
Package: gov.va.wss.form686.services.service.pdf

**PDFHeaderRequest**

* byte[] bytes
* Long submissionDate
* String confirmationNumber
* String wssRole
* Long creationDate

**PDFManagmentResponse**

* byte[] pdf



**PDFManagementServiceImpl**

Method:make686Pdf

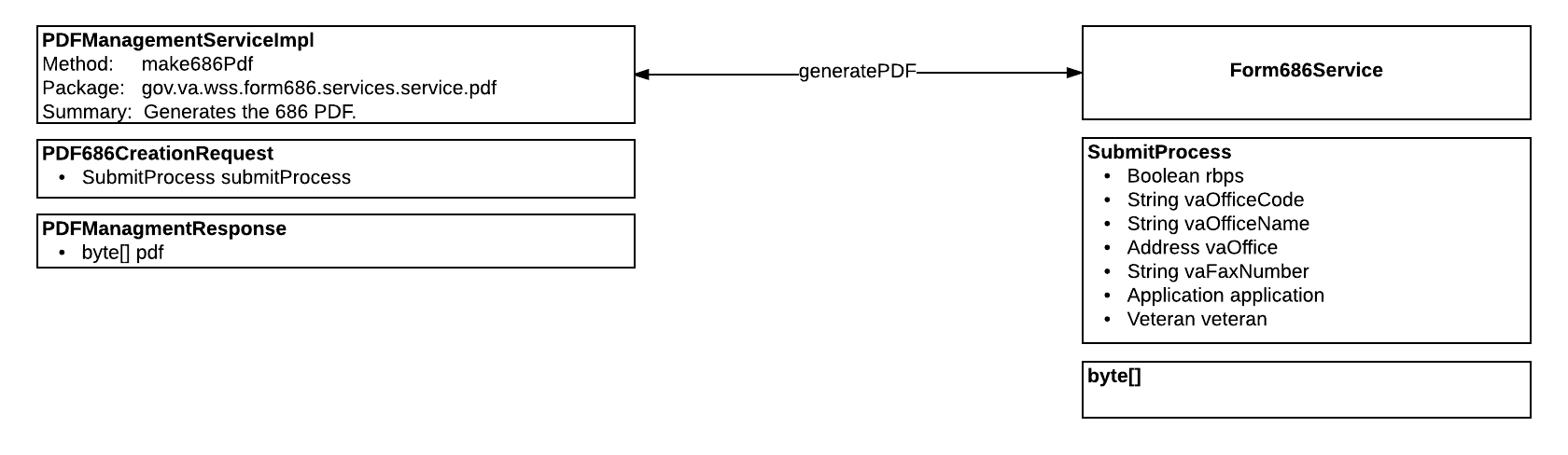
Package: gov.va.wss.form686.services.service.pdf

**PDF686CreationRequest**

* SubmitProcess submitProcess

**PDFManagmentResponse**

* byte[] pdf



**PDFManagementServiceImpl**

Method:make674PDF

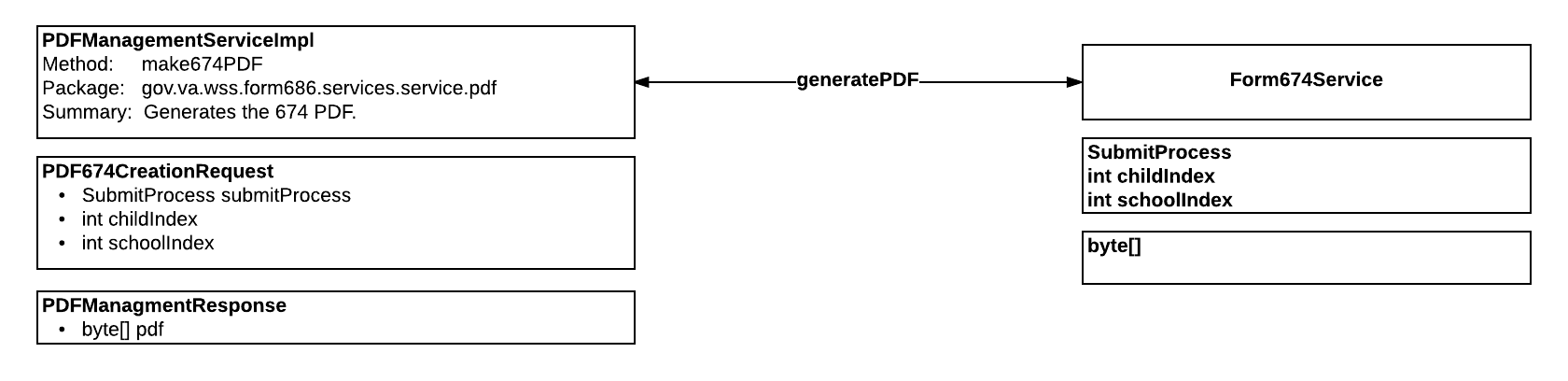
Package: gov.va.wss.form686.services.service.pdf

**PDF674CreationRequest**

* SubmitProcess submitProcess
* int childIndex
* int schoolIndex

**PDFManagmentResponse**

* byte[] pdf



**PDFManagementServiceImpl**

Method:getCoverSheet

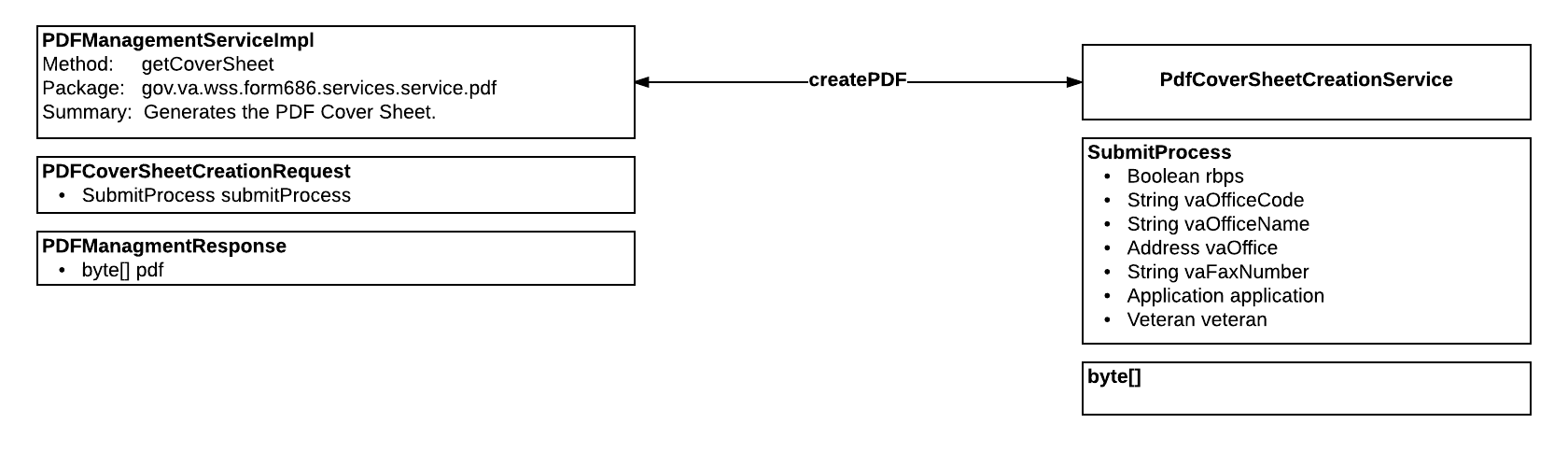
Package: gov.va.wss.form686.services.service.pdf

**PDFCoverSheetCreationRequest**

* SubmitProcess submitProcess

**PDFManagmentResponse**

* byte[] pdf



### PdfHeaderCreationService

The purpose of this service is to generate the PDF header.

**PdfHeaderCreationServiceImpl**

Method:generateHeaderResponse

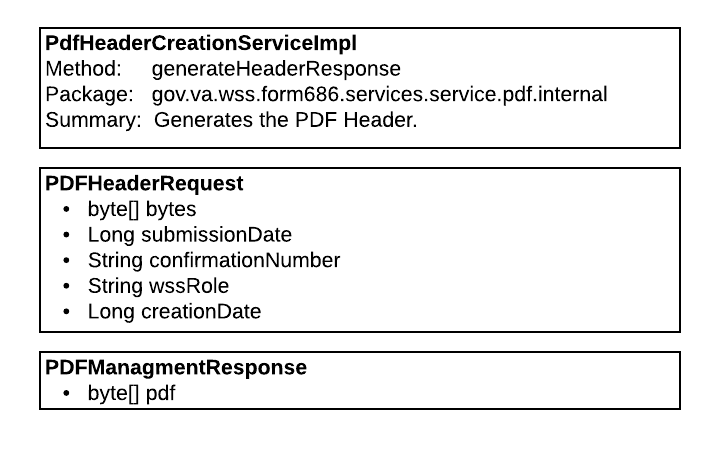
Package: gov.va.wss.form686.services.service.pdf.internal

**PDFHeaderRequest**

* byte[] bytes
* Long submissionDate
* String confirmationNumber
* String wssRole
* Long creationDate

**PDFManagmentResponse**

* byte[] pdf



### Form686Service

The purpose of this service is to generate the 686 PDF.

**Form686ServiceImpl**

Method:generatePDF

Package: gov.va.wss.form686.services.service.pdf.internal

**SubmitProcess**

Boolean rbps

String vaOfficeCode

String vaOfficeName

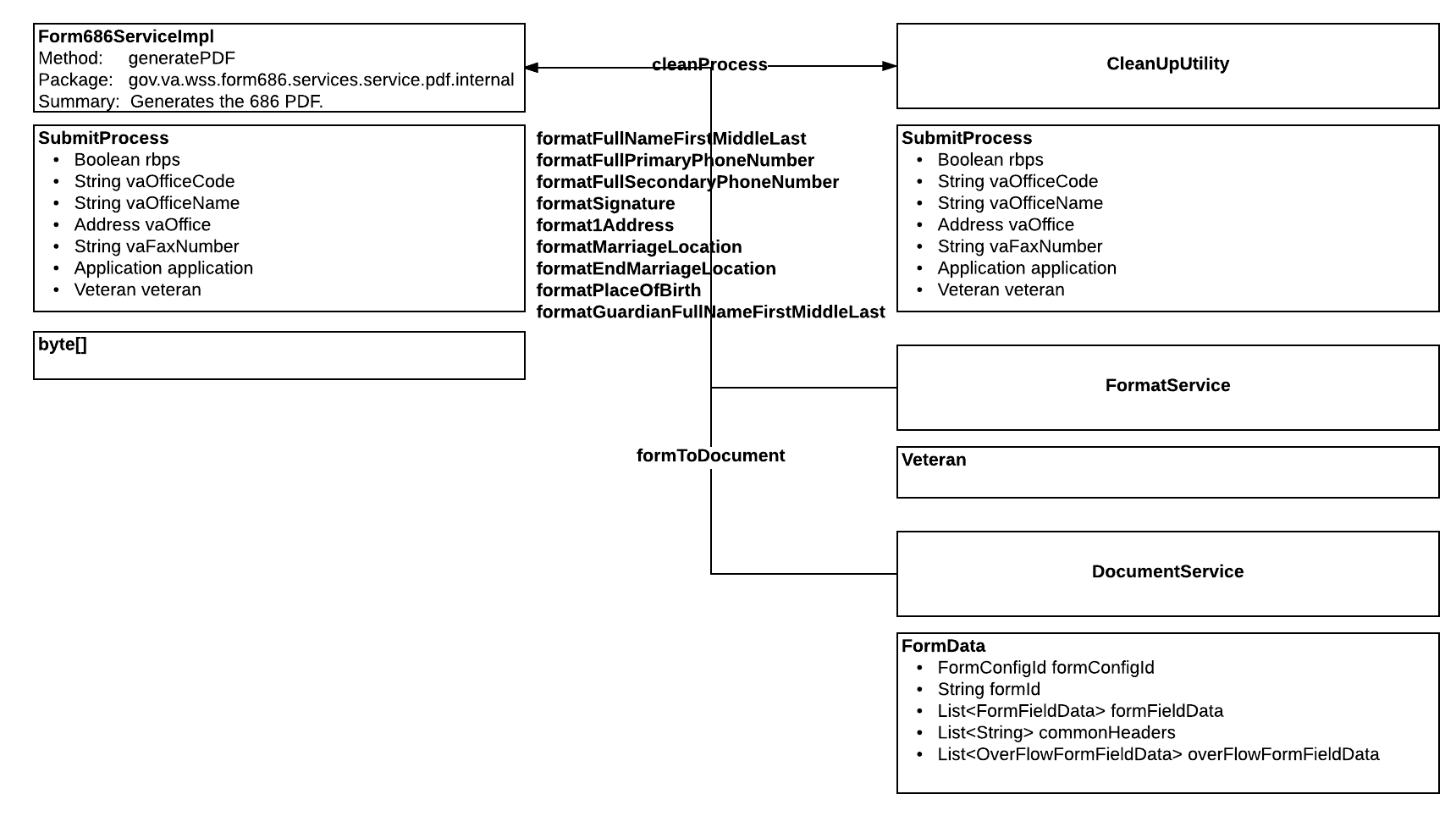
Address vaOffice

String vaFaxNumber

Application application

Veteran veteran

**byte[]**



### Form674Service

The purpose of this service is to generate the 674 PDF.

**Form674ServiceImpl**

Method: generatePDF

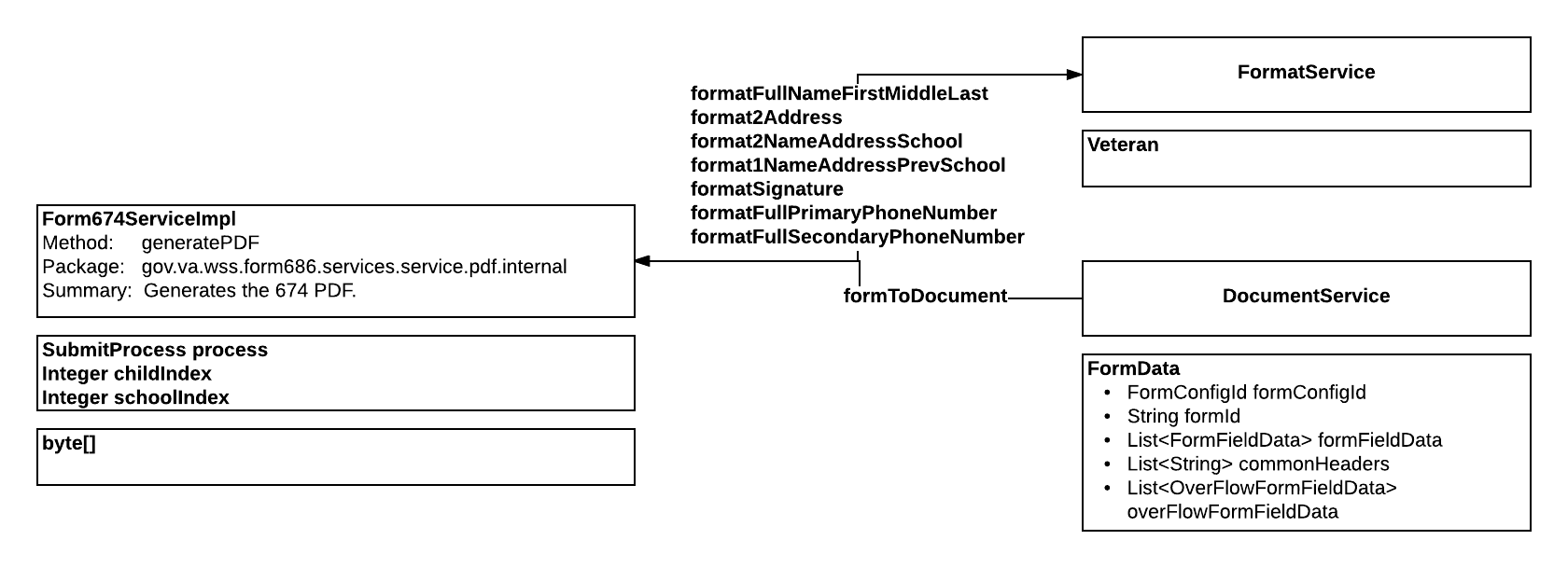
Package: gov.va.wss.form686.services.service.pdf.internal

**SubmitProcess**

**Integer**

**Integer**

**byte[]**



### PdfCoverSheetCreationService

The purpose of this service is to generate the PDF cover sheet.

**PdfCoverSheetCreationServiceImpl**

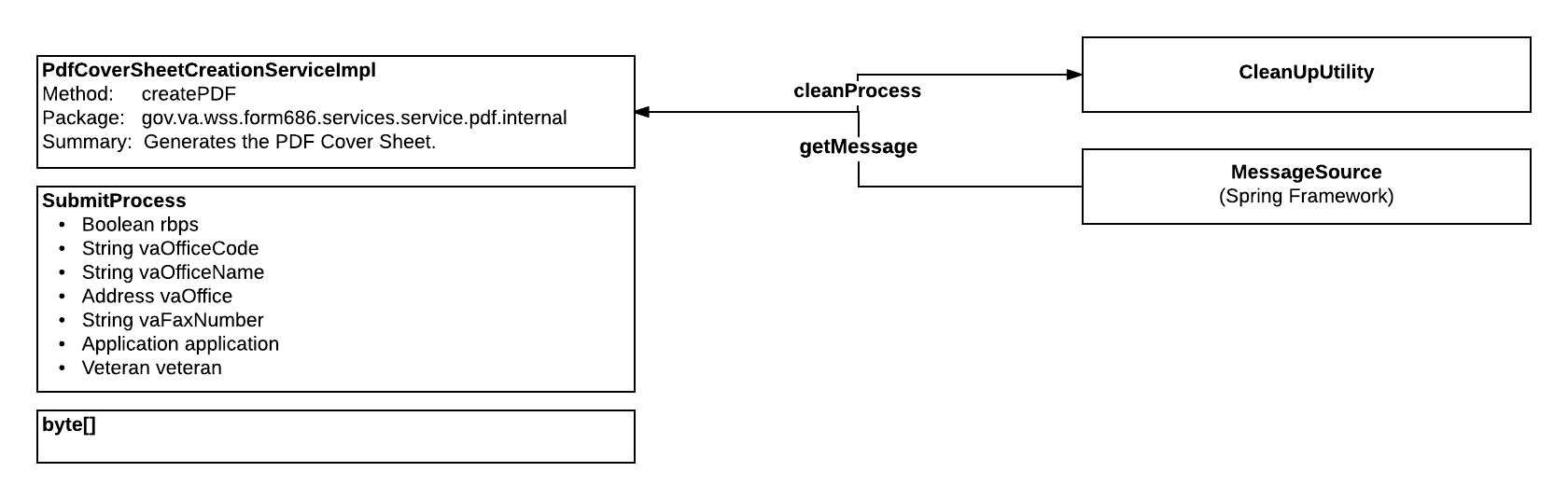
 Method: createPDF

Package: gov.va.wss.form686.services.service.pdf.internal

**SubmitProcess**

* Boolean rbps
* String vaOfficeCode
* String vaOfficeName
* Address vaOffice
* String vaFaxNumber
* Application application
* Veteran veteran

**byte[]**



### SchoolLookUpService

The purpose of this service is to do the following:

* Find schools by name or facility code. The service allows the caller to limit the number of results returned
* Retrieve school details by Participant ID and Facility Code
* Retrieve the State Codes in a Map

**SchoolLookUpServiceImpl**

Method: search

Package: gov.va.wss.form686.services.service.school

**SchoolSearchRequest**

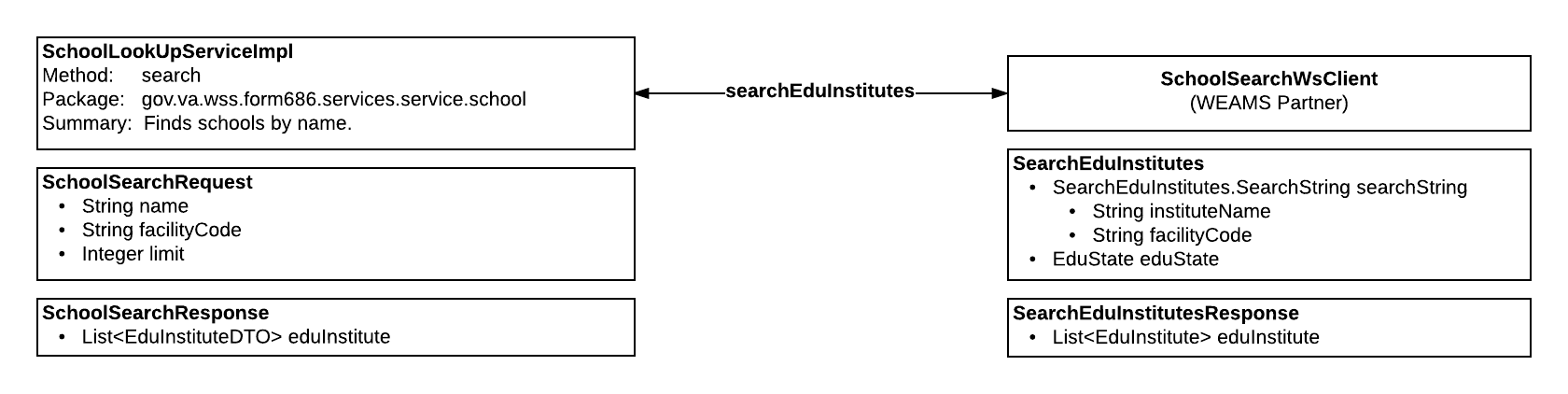
String name

String facilityCode

Integer limit

**SchoolSearchResponse**

List<EduInstituteDTO> eduInstitute



**SchoolLookUpServiceImpl**

 Method: getDetails

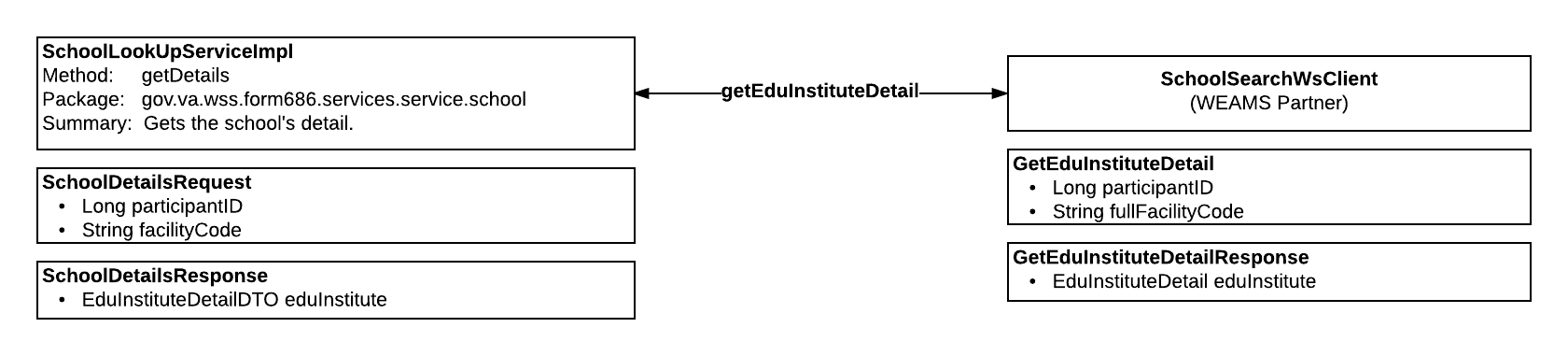
Package: gov.va.wss.form686.services.service.school

**SchoolDetailsRequest**

* Long participantID
* String facilityCode

**SchoolDetailsResponse**

* EduInstituteDetailDTO eduInstitute



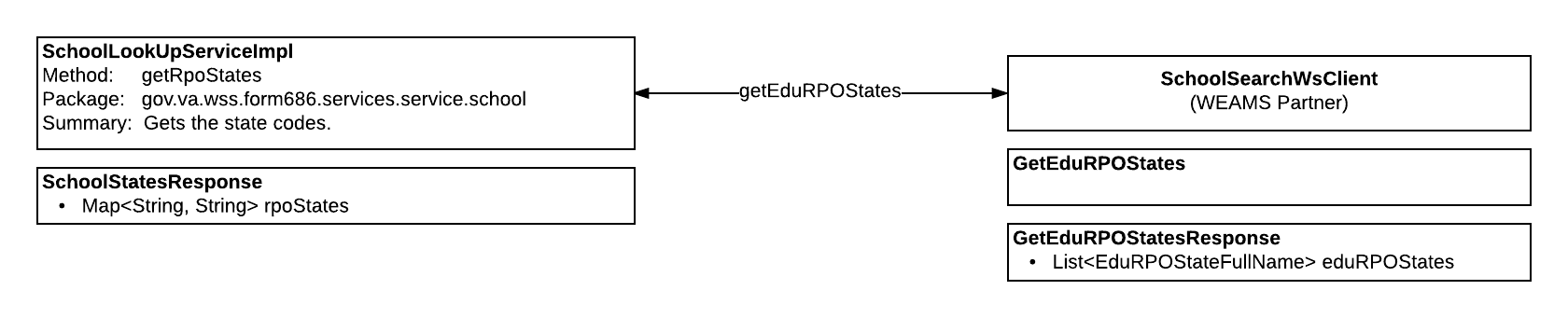
**SchoolLookUpServiceImpl**

Method: getRpoStates

Package: gov.va.wss.form686.services.service.school

**SchoolStatesResponse**

* Map<String, String> rpoStates



### Form686SubmissionService

The purpose of this service is to do the following:

* Submits the 686 Form
  + Inserts the Benefit Claim
  + Saves the form to InFlight
  + Save an audit record to the   User Process Mapper table
  + Saves an Audit   Status Change
  + Runs the Batch Job.

**Form686SubmissionServiceImpl**

Method: submit686

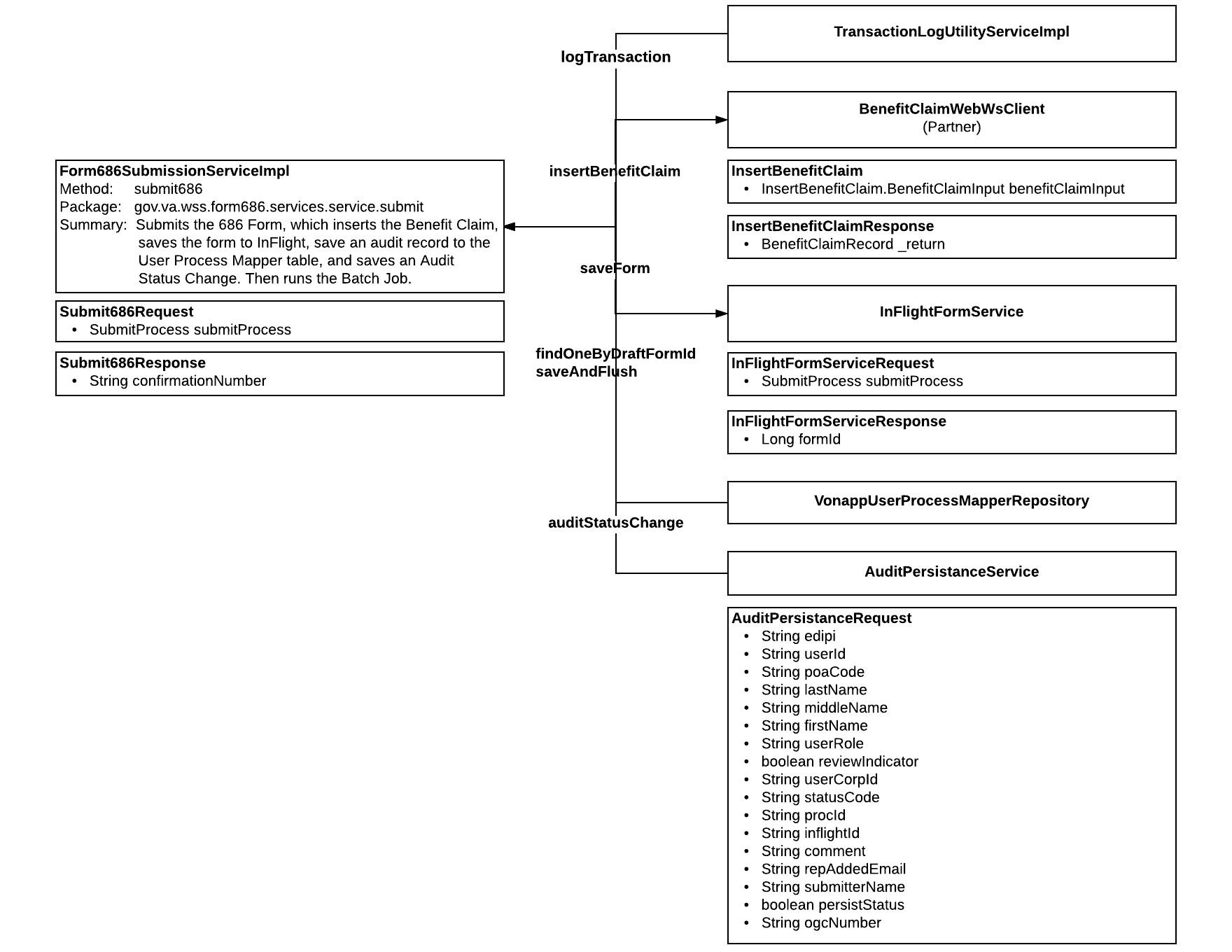
Package: gov.va.wss.form686.services.service.submit

**Submit686Request**

* SubmitProcess submitProcess

**Submit686Response**

* String confirmationNumber



### SyncService

The purpose of this service is to sync and save dependents and spouses.

**SyncServiceImpl**

Method: syncDependents

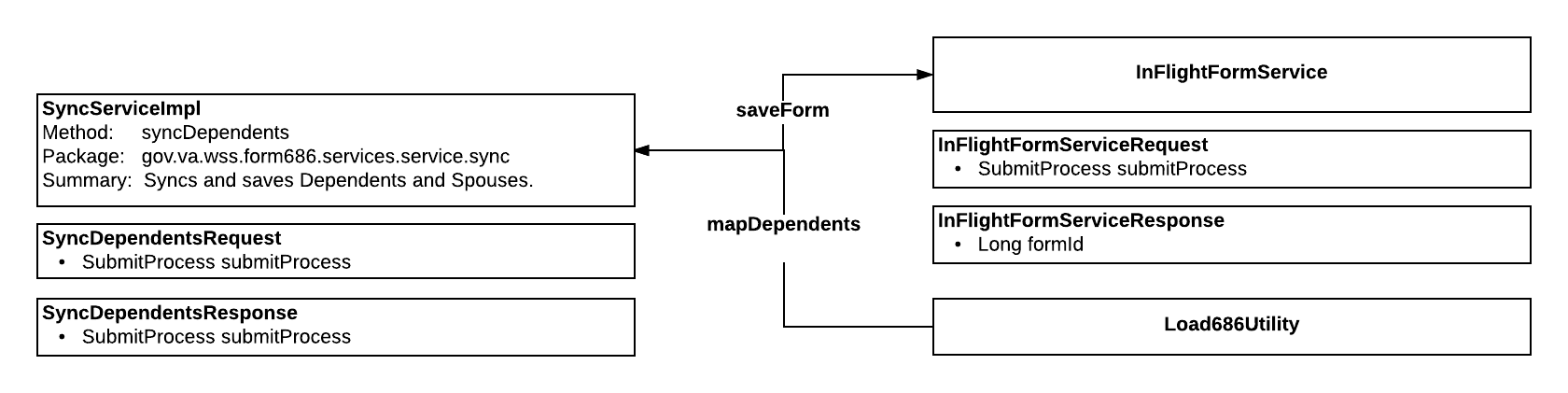
Package: gov.va.wss.form686.services.service.sync

**SyncDependentsRequest**

* SubmitProcess submitProcess

**SyncDependentsResponse**

* SubmitProcess submitProcess



### TransactionLogUtilityService

The purpose of this service is to save a log of the transaction to the database.

**TransactionLogUtilityServiceImpl**

Method: logTransaction

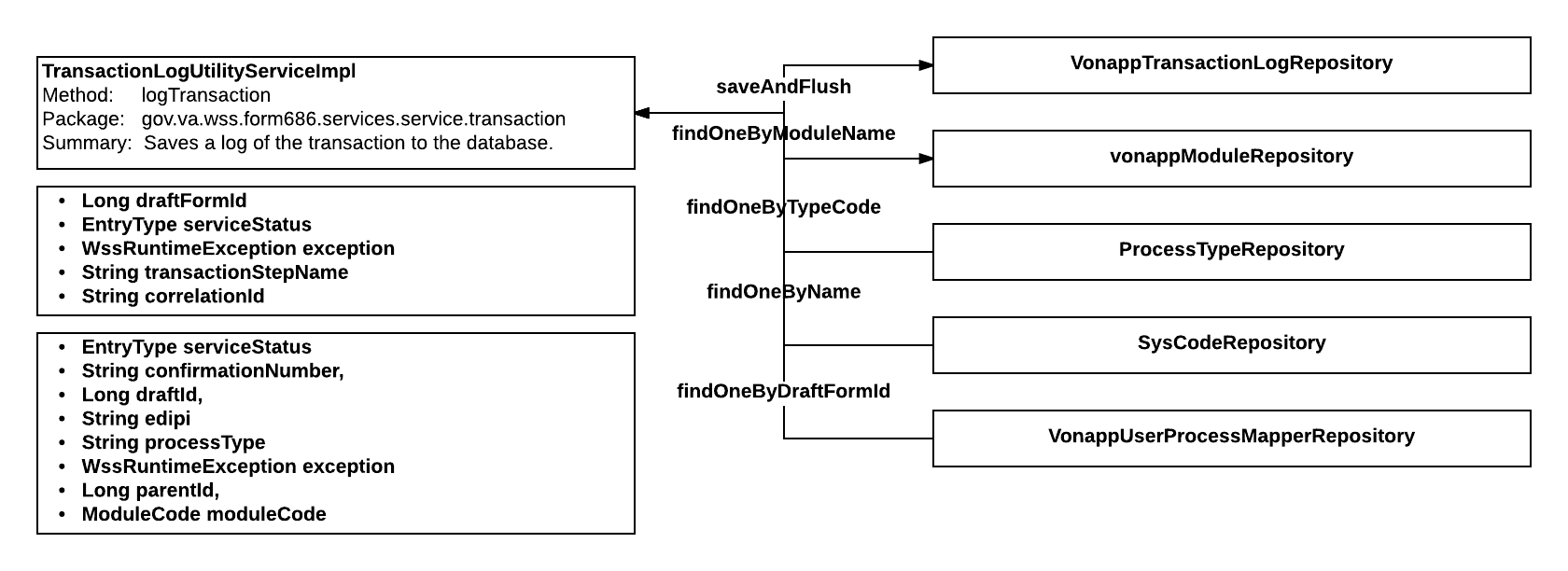
Package: gov.va.wss.form686.services.service.transaction

**Method 1**

* Long draftFormId
* EntryType serviceStatus
* WssRuntimeException exception
* String transactionStepName
* String correlationId

**Method 2**

* EntryType serviceStatus
* String confirmationNumber
* Long draftId
* String edipi
* String processType
* WssRuntimeException exception
* Long parentId
* ModuleCode moduleCode

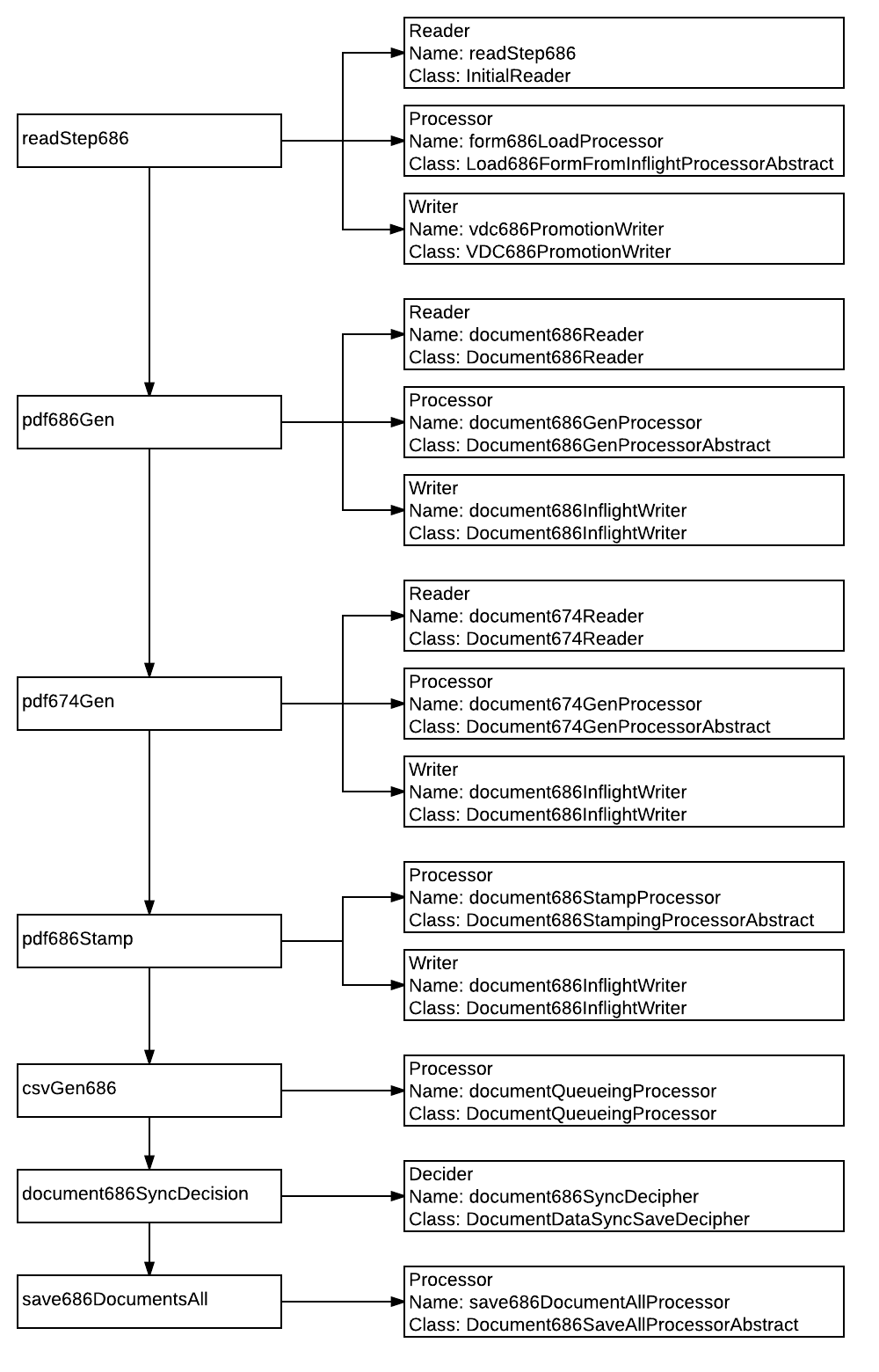


## Batch Jobs

The following are the batch jobs that exist in wss-686-services today. The following sections break the batch jobs down by steps and a flow diagram.

### Document 686 Job

This job is responsible for pulling the 686 and 674 forms from the database, generating the PDFs, and sending them to external partner services for storage.



#### readStep686

This step pulls all of the 686/674 information and saves it to the Cache.

**Reader**

Name: readStep686

Class: InitialReader

**Processor**

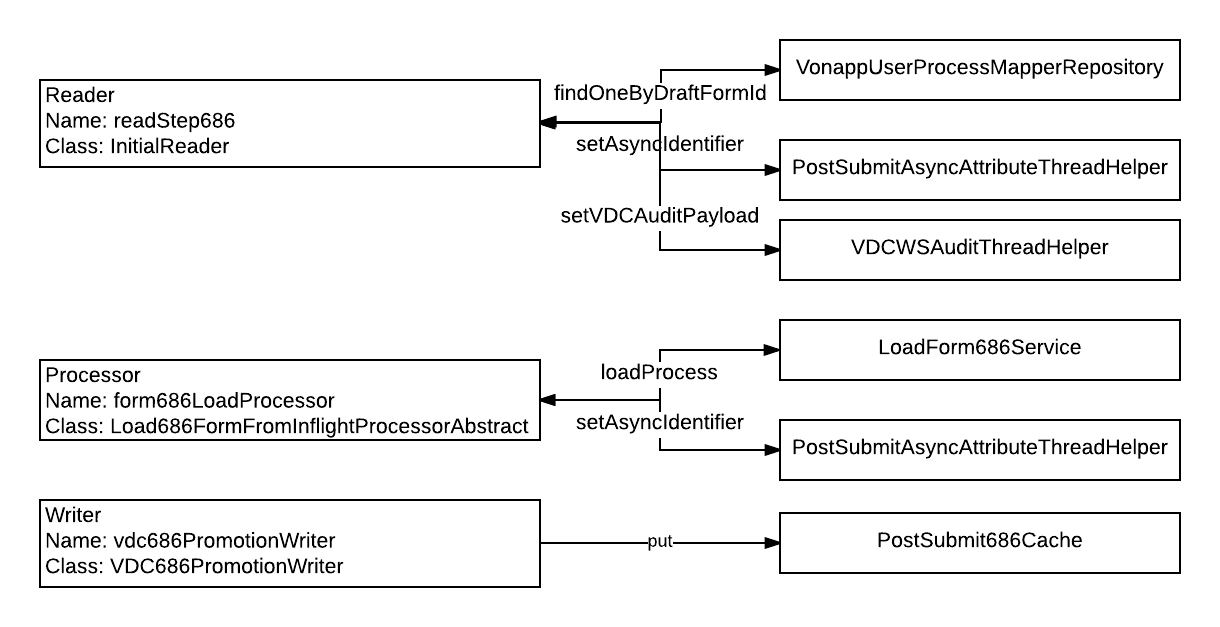
Name: form686LoadProcessor

Class: Load686FormFromInflightProcessorAbstract

**Writer**

Name: vdc686PromotionWriter

Class: VDC686PromotionWriter



#### pdf686Gen

These steps take the information in the Cache and generates the 686 PDF forms, then saves the forms using the VRMDraftFormWsClient. It also logs the event in the Vonapp User Process Mapper table.

**Reader**

Name: document686Reader

Class: Document686Reader

**Processor**

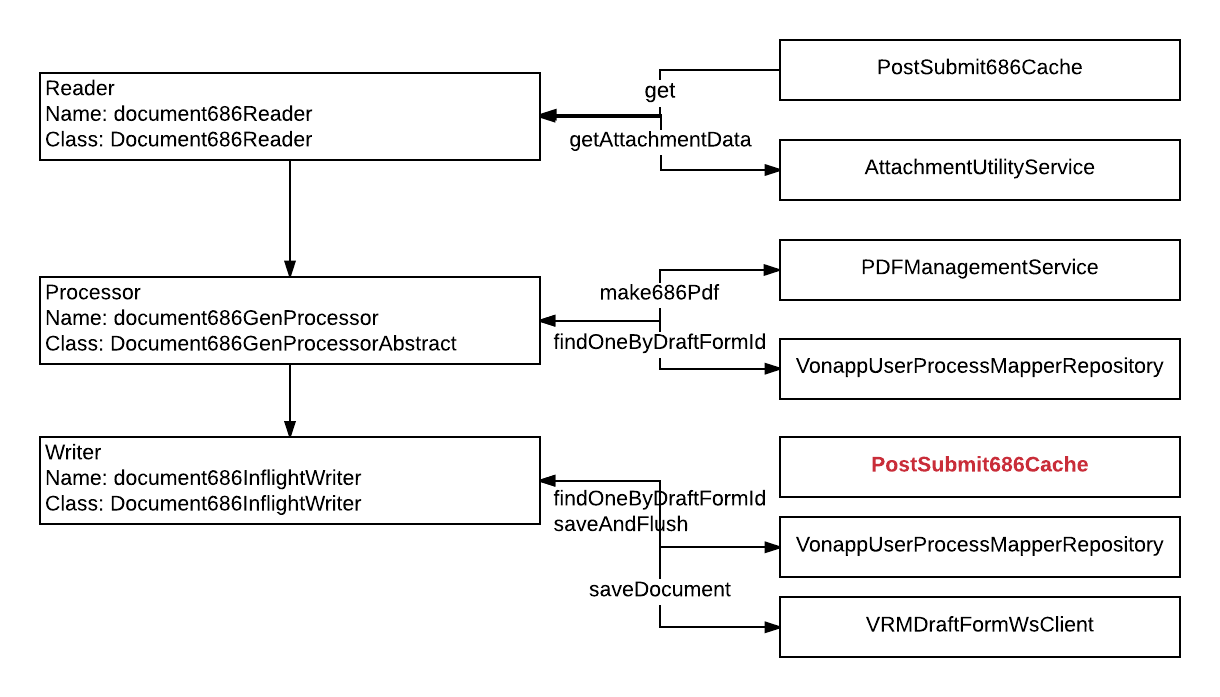
Name: document686GenProcessor

Class: Document686GenProcessorAbstract

**Writer**

Name: document686InflightWriter

Class: Document686InflightWriter



#### pdf674Gen

This steps takes the information in the Cache and generates the 674 PDF forms, then saves the forms using the VRMDraftFormWsClient. It also logs the event in the Vonapp User Process Mapper table.

**Reader**

Name: document674Reader

Class: Document674Reader

**Processor**

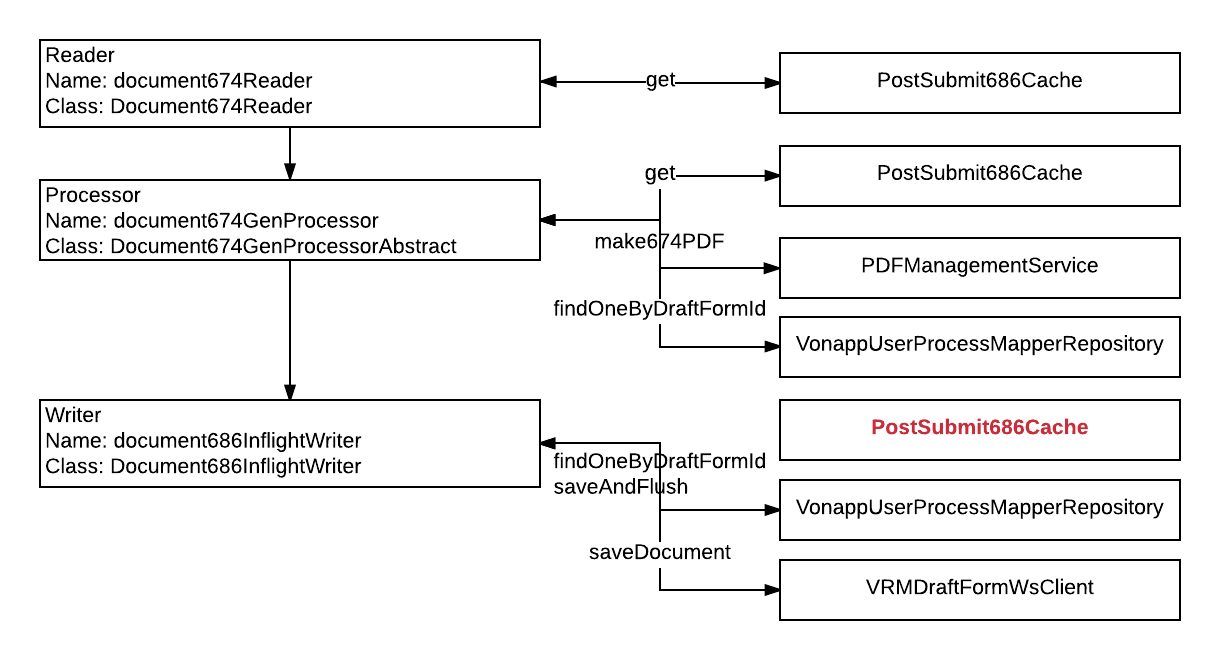
Name: document674GenProcessor

Class: Document674GenProcessorAbstract

**Writer**

Name: document686InflightWriter

Class: Document686InflightWriter



#### pdf686Stamp

These steps take the PDF forms and adds a header (stamp), then saves the forms using the VRMDraftFormWsClient. It also logs the event in the Vonapp User Process Mapper table.

**Processor**

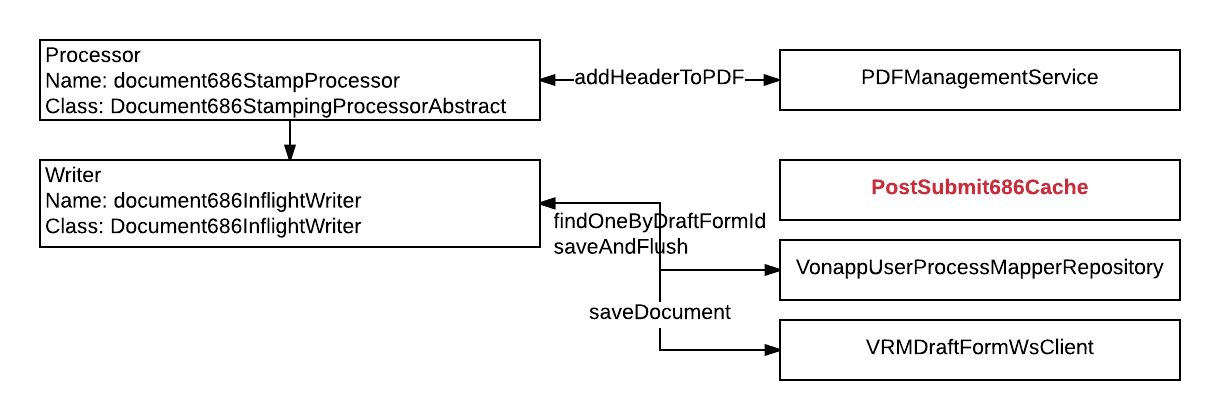
Name: document686StampProcessor

Class: Document686StampingProcessorAbstract

**Writer**

Name: document686InflightWriter

Class: Document686InflightWriter



#### csvGen686

These steps call the QueuedDocumentTransferExternalService to add the Staged Document to the Queue.

**Processor**

Name: documentQueueingProcessor

Class: DocumentQueueingProcessor



#### document686SyncDecision

This step is a decider that determines whether the call save686DocumentsAll.

**Decider**

Name: document686SyncDecipher

Class: DocumentDataSyncSaveDecipher



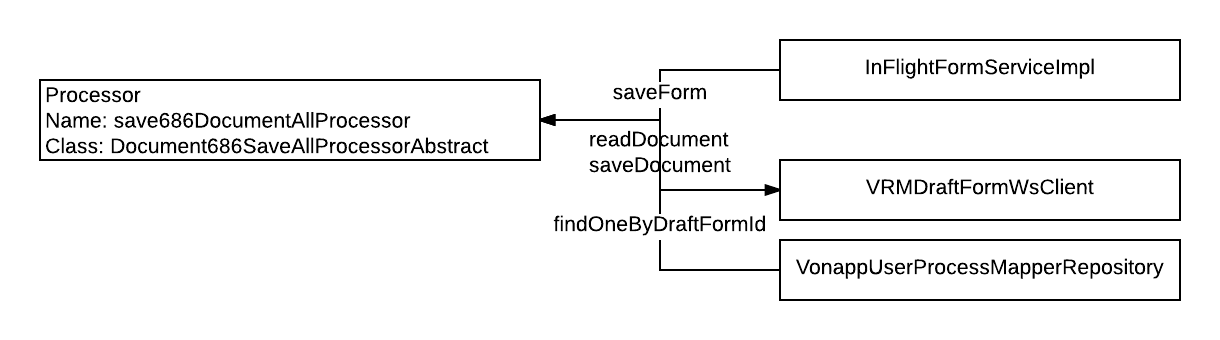
#### save686DocumentsAll

This step saves the form/document and the event to InFlight, VRMDraftForm, and Vonapp User Process Mapper table.

**Processor**

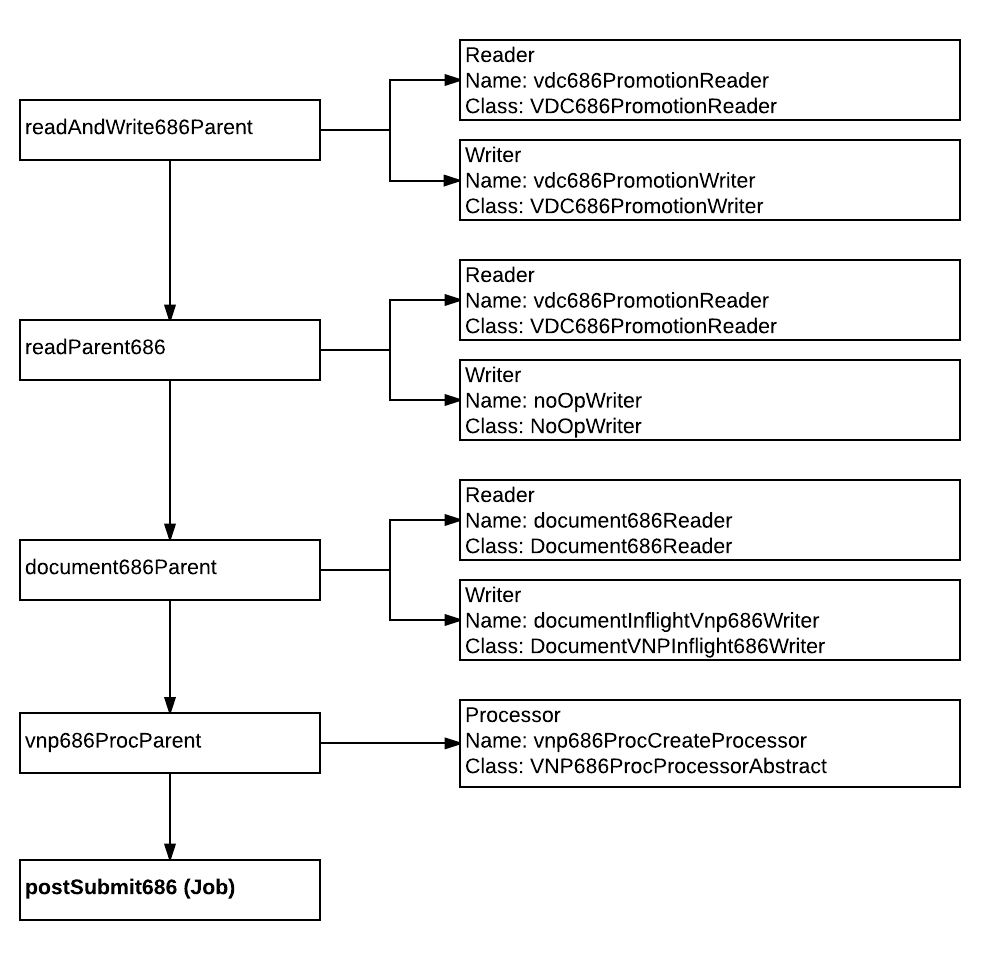
Name: save686DocumentAllProcessor

Class: Document686SaveAllProcessorAbstract



### Submit 686 Job

This job controls the actual submission of the 686/674 forms.



#### readAndWrite686Parent

This step offers convenience methods for reading information from the Cache, then putting it back in the Cache.

**Reader**

Name: vdc686PromotionReader

Class: VDC686PromotionReader

**Writer**

Name: vdc686PromotionWriter

Class: VDC686PromotionWriter



#### 

#### readParent686

This step reads information from the Cache.

**Reader**

Name: vdc686PromotionReader

Class: VDC686PromotionReader

**Writer**

Name: noOpWriter

Class: NoOpWriter



#### document686Parent

This parent step provides convenience methods for retrieving and saving documents and attachments.

**Reader**

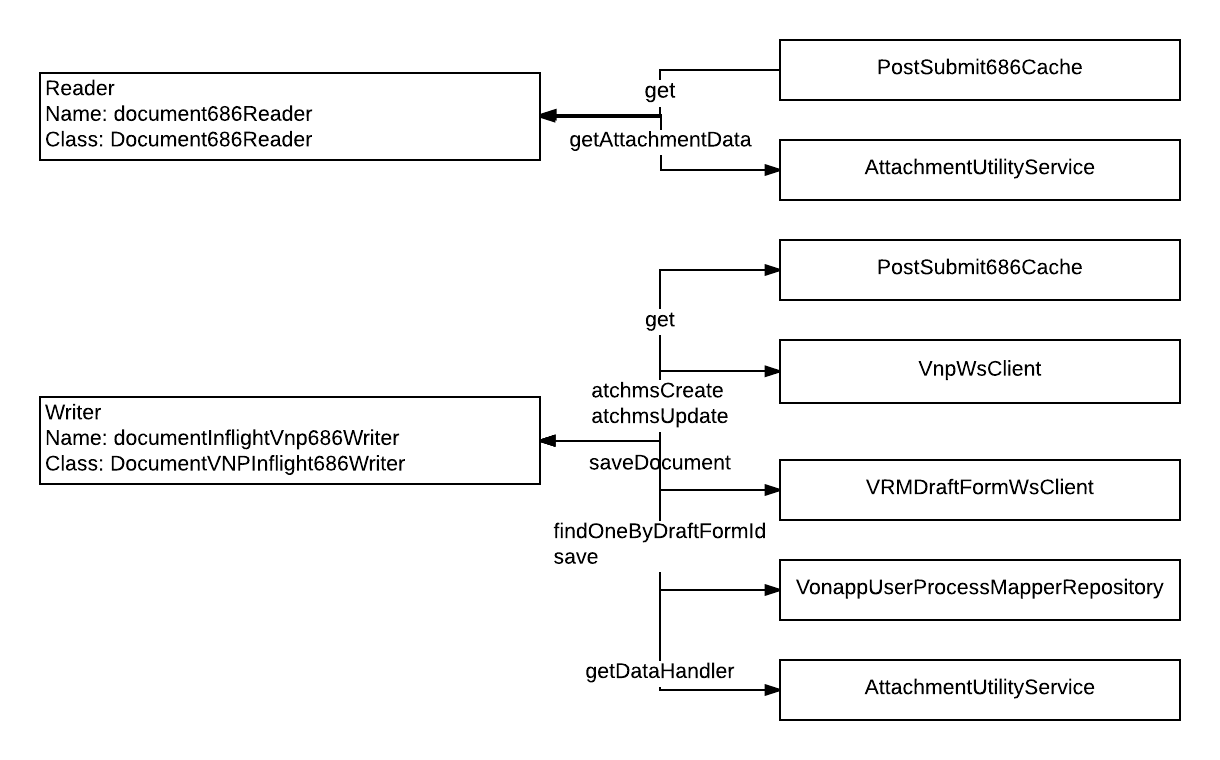
Name: document686Reader

Class: Document686Reader

**Writer**

Name: documentInflightVnp686Writer

Class: DocumentVNPInflight686Writer



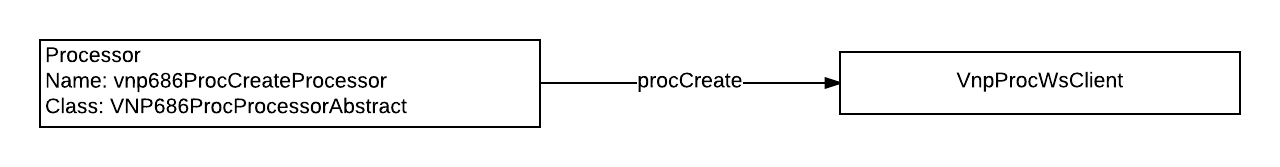
#### vnp686ProcParent

This step creates the 686 Process.

**Processor**

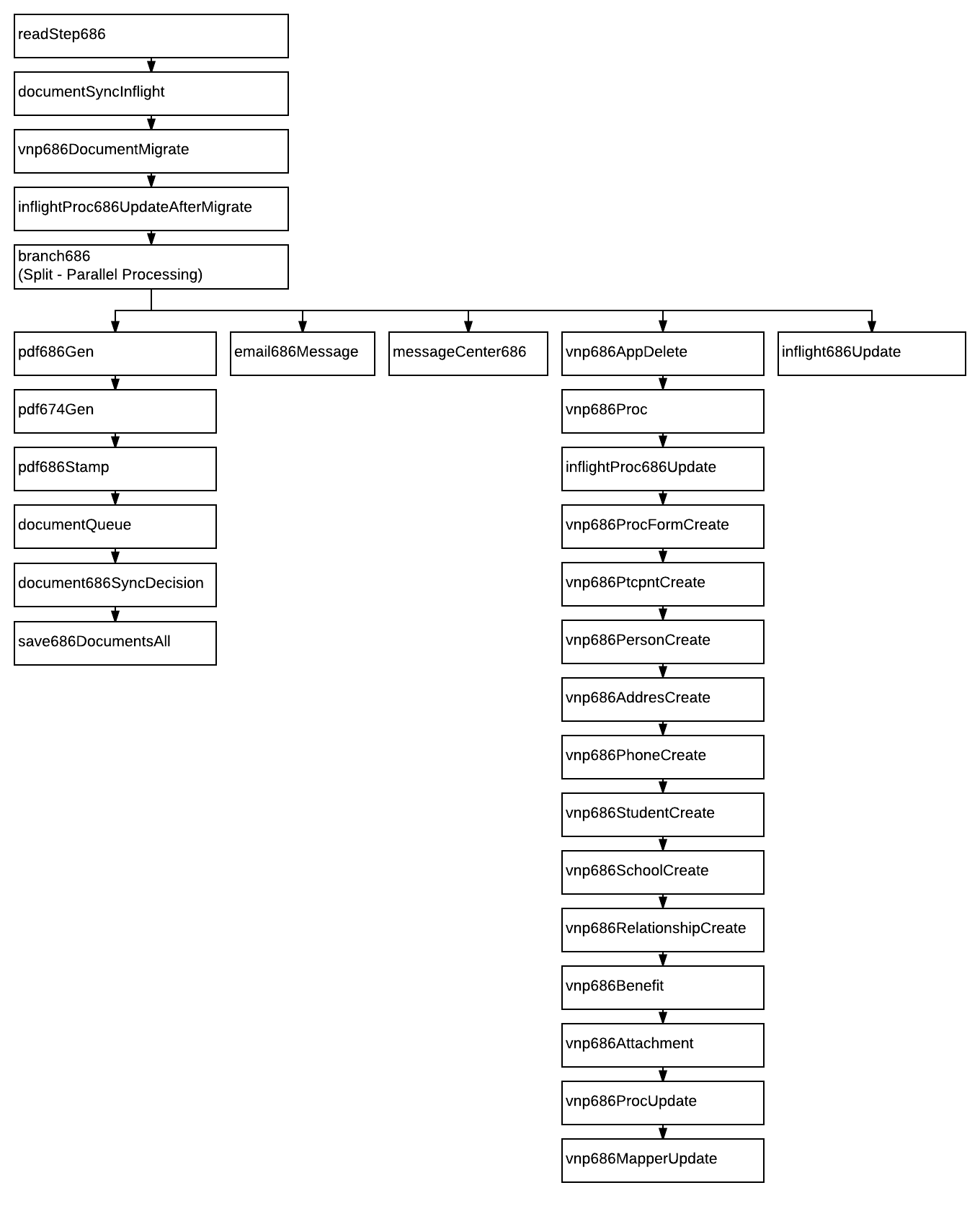
Name: vnp686ProcCreateProcessor

Class: VNP686ProcProcessorAbstract



### Post Submit 686 Job

This is a child job inside the previous job. This job breaks into parallel processing.



#### readStep686

This step pulls all of the 686/674 information from the various data stores and compiles it for processing.

**Reader**

Name: intitialReader

Class: InitialReader

**Processor**

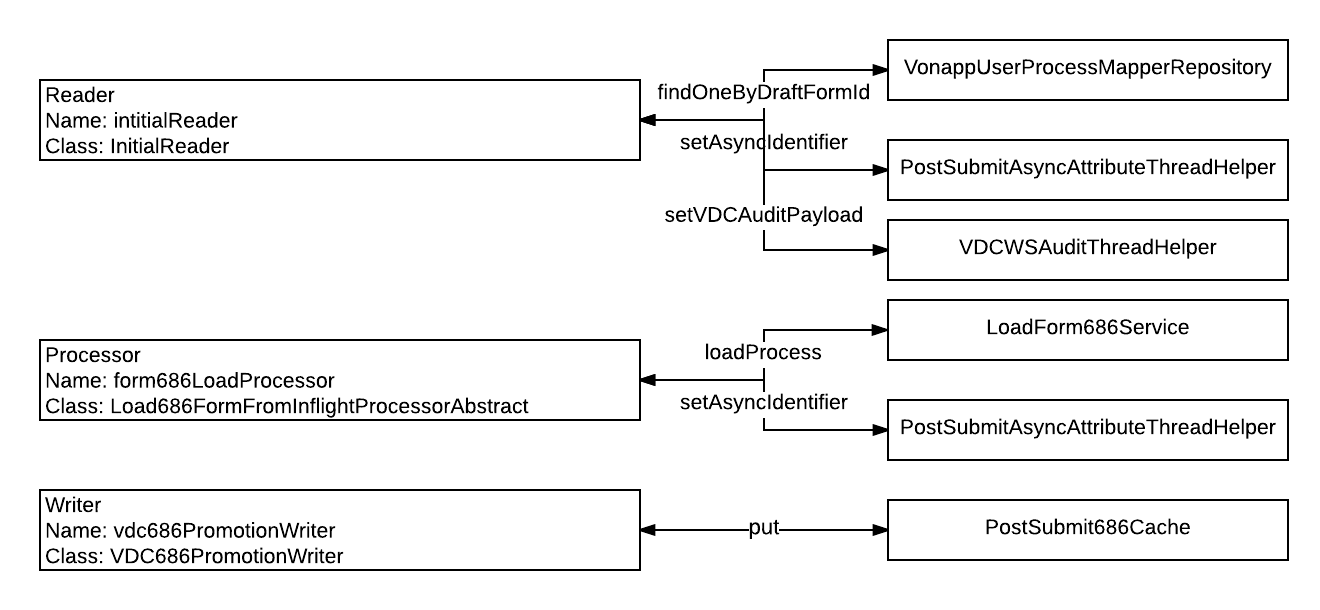
Name: form686LoadProcessor

Class: Load686FormFromInflightProcessorAbstract

**Writer**

Name: vdc686PromotionWriter

Class: VDC686PromotionWriter



#### documentSyncInflight

This step deletes the documents from InFlight.

**Reader**

Name: document686DBSyncReader

Class: DocumentDBSyncReader

**Processor**

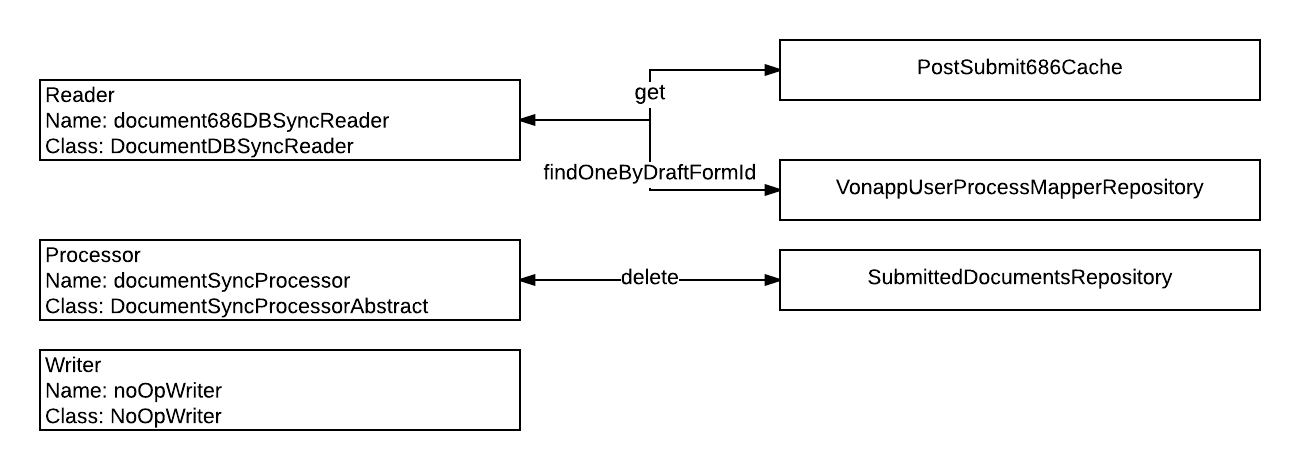
Name: documentSyncProcessor

Class: DocumentSyncProcessorAbstract

**Writer**

Name: noOpWriter

Class: NoOpWriter



#### vnp686DocumentMigrate

This step sends the documents to VRMDraftForm and VNP. It also records the event in the Vonapp User Process Mapper.

**Reader**

Name: document686VnpReader

Class: VNPDocument686Reader

**Processor**

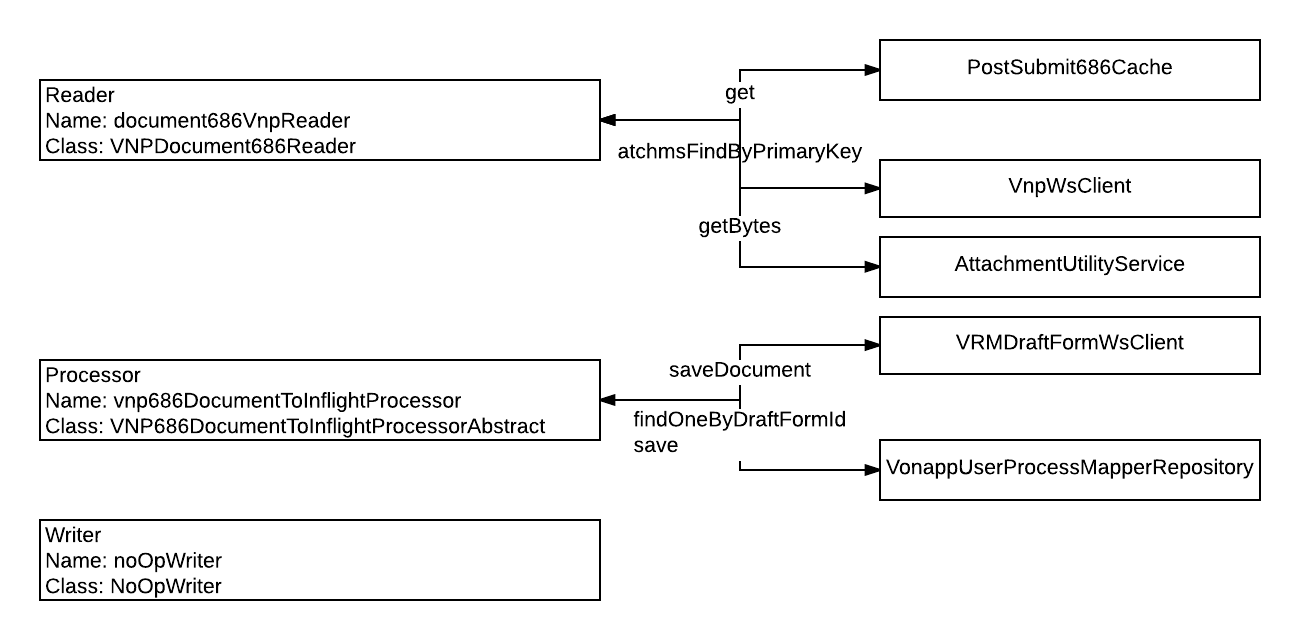
Name: vnp686DocumentToInflightProcessor

Class: VNP686DocumentToInflightProcessorAbstract

**Writer**

Name: noOpWriter

Class: NoOpWriter



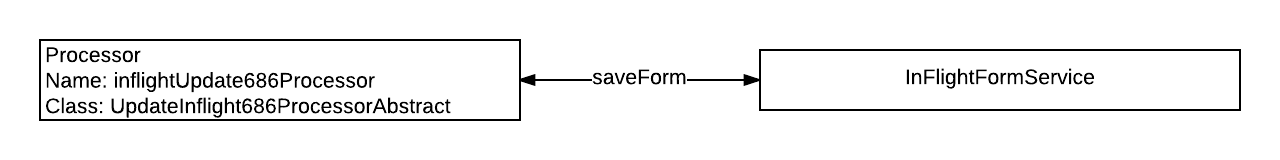
#### inflightProc686UpdateAfterMigrate

This step saves the Form back to InFlight after migration.

**Processor**

Name: inflightUpdate686Processor

Class: UpdateInflight686ProcessorAbstract



#### pdf686Gen

This step generates the 686 PDF. Saves the form to VRMDraftForm and the event to Vonapp User Process Mapper table.

**Reader**

Name: document686Reader

Class: Document686Reader

**Processor**

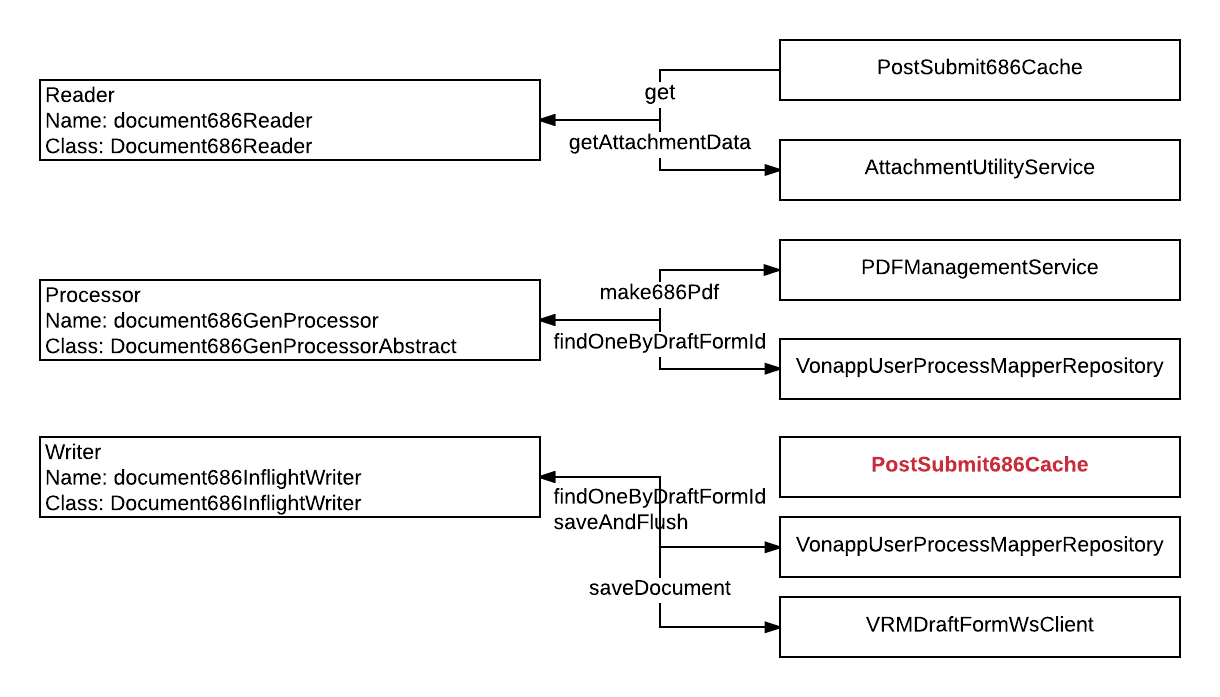
Name: document686GenProcessor

Class: Document686GenProcessorAbstract

**Writer**

Name: document686InflightWriter

Class: Document686InflightWriter



#### pdf674Gen

This step generates the 674 PDF. Saves the form to VRMDraftForm and the event to Vonapp User Process Mapper table.

**Reader**

Name: document674Reader

Class: Document674Reader

**Processor**

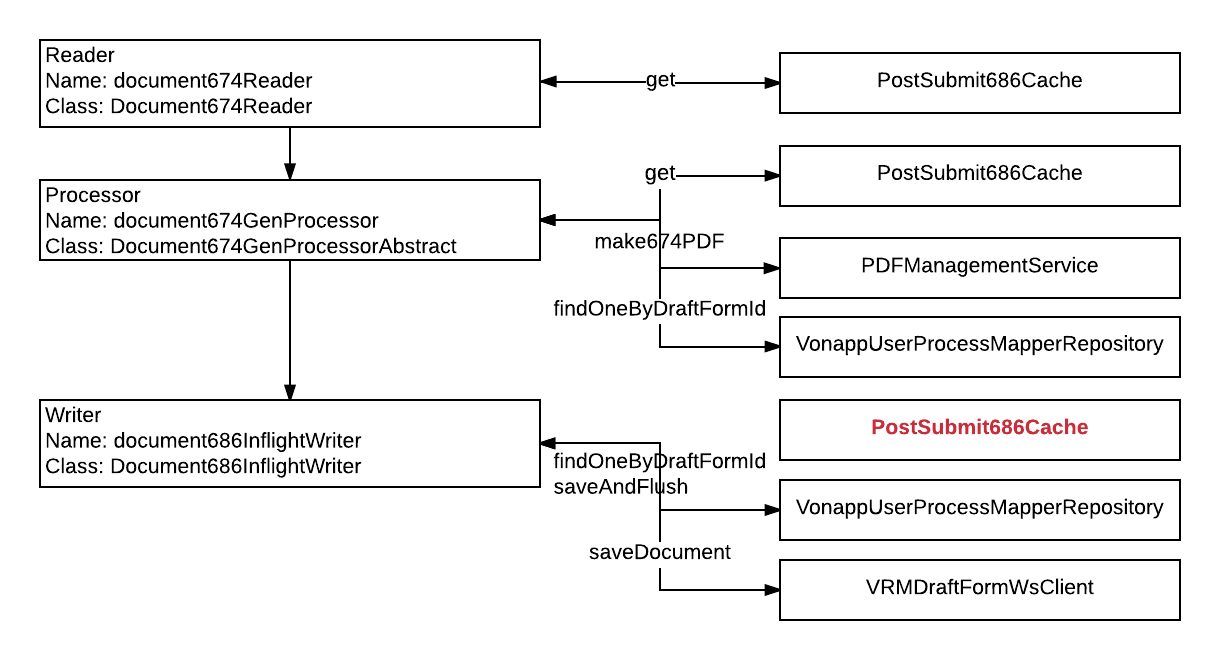
Name: document674GenProcessor

Class: Document674GenProcessorAbstract

**Writer**

Name: document686InflightWriter

Class: Document686InflightWriter



#### pdf686Stamp

This step adds the PDF Header to the Document. Then saves the Form to VRMDraftForm and the event to the Vonapp User Process Mapper table.

**Processor**

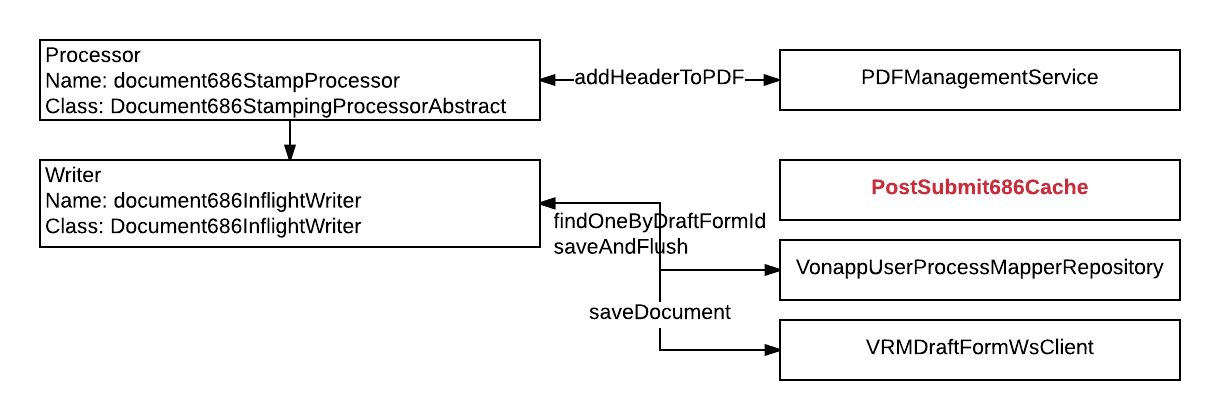
Name: document686StampProcessor

Class: Document686StampingProcessorAbstract

**Writer**

Name: document686InflightWriter

Class: Document686InflightWriter



#### documentQueue

This step adds the Staged Document to the Queue.

**Reader**

Name: documentQueueing686VnpReader

Class: VNPDocumentQueuing686Reader

**Processor**

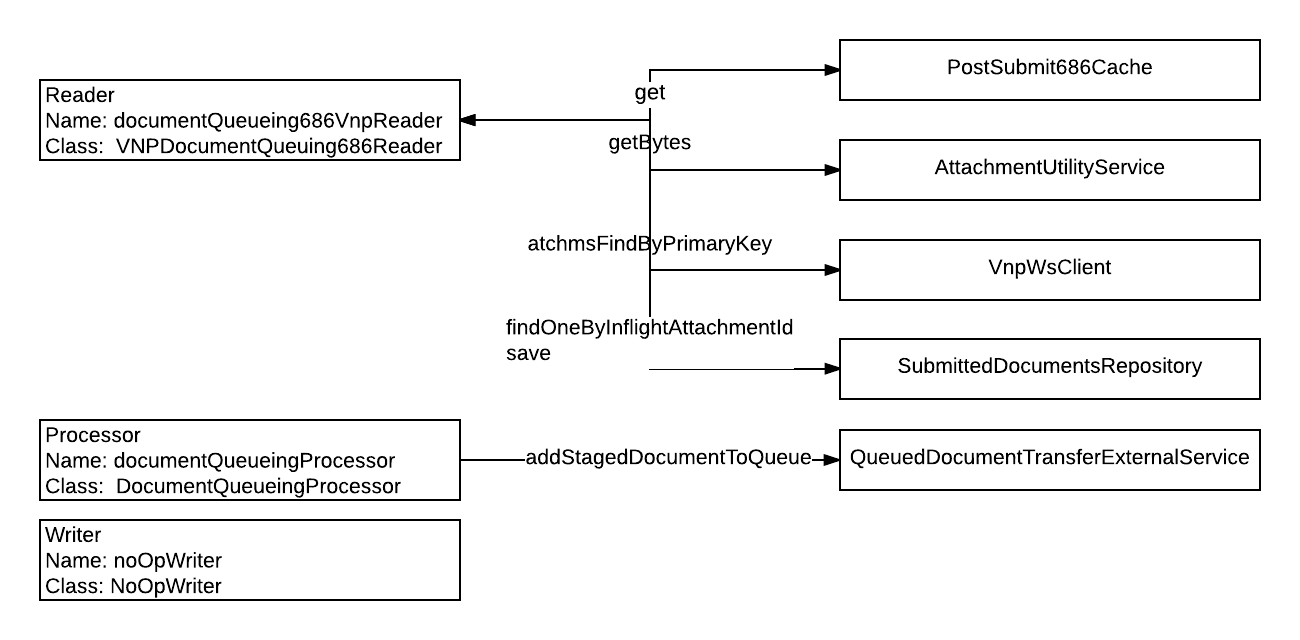
Name: documentQueueingProcessor

Class: DocumentQueueingProcessor

**Writer**

Name: noOpWriter

Class: NoOpWriter



#### document686SyncDecision

This step decides whether to call save686DocumentsAll.

**Decider**

Name: document686SyncDecipher

Class: DocumentDataSyncSaveDecipher



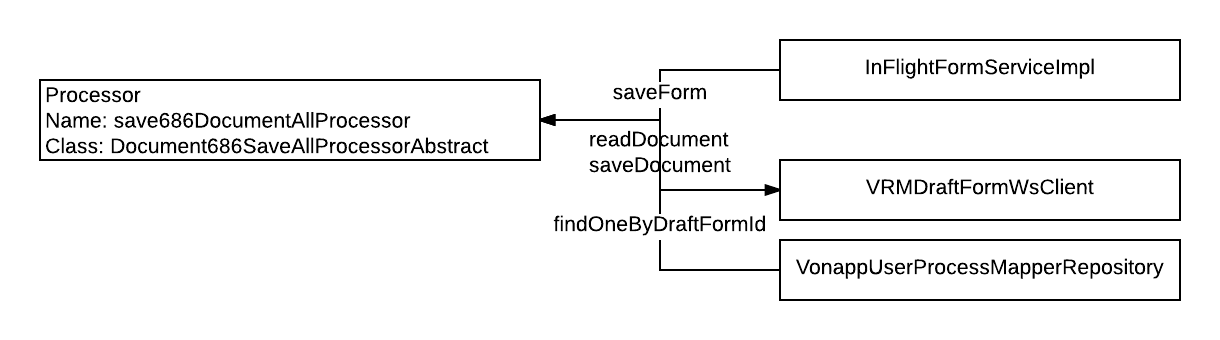
#### save686DocumentsAll

This step saves the Forms/Documents to InFlight and VRMDraftForm, and the event to the Vonapp User Process Mapper.

**Processor**

Name: save686DocumentAllProcessor

Class: Document686SaveAllProcessorAbstract



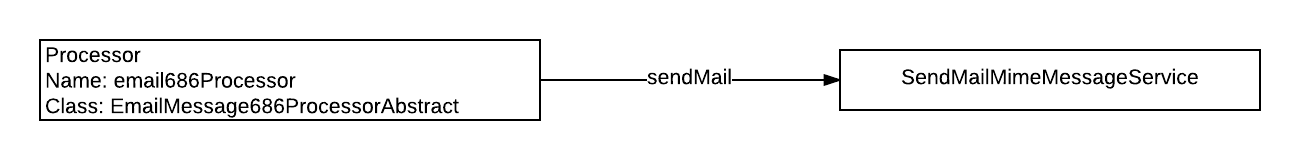
#### email686Message

This step occurs in parallel and sends an email regarding the success/failure of the batch job.

**Processor**

Name: email686Processor

Class: EmailMessage686ProcessorAbstract



#### messageCenter686

This step occurs in parallel and sends User Messages regarding the success/failure of the batch job.

**Reader**

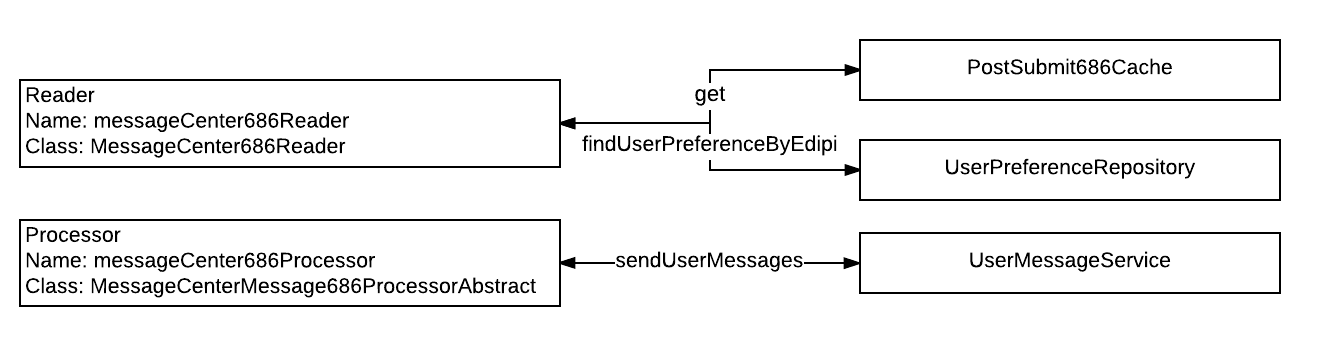
Name: messageCenter686Reader

Class: MessageCenter686Reader

**Processor**

Name: messageCenter686Processor

Class: MessageCenterMessage686ProcessorAbstract



#### vnp686AppDelete

This step starts another flow that runs in parallel with the first flow in the split. This step removes the Vnp Process.

**Reader**

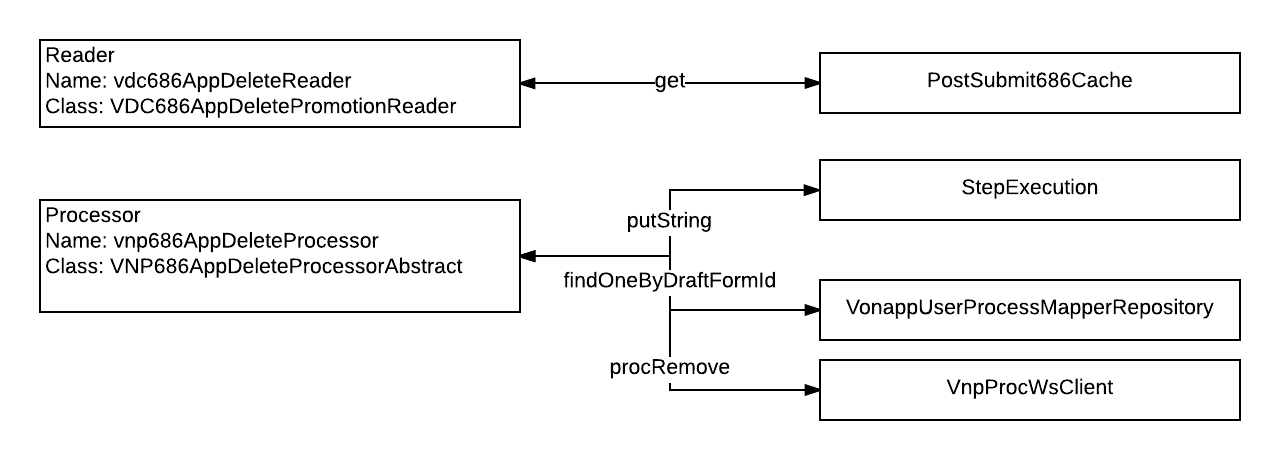
Name: vdc686AppDeleteReader

Class: VDC686AppDeletePromotionReader

**Processor**

Name: vnp686AppDeleteProcessor

Class: VNP686AppDeleteProcessorAbstract



#### vnp686Proc

There is nothing defined in this step.

N/A

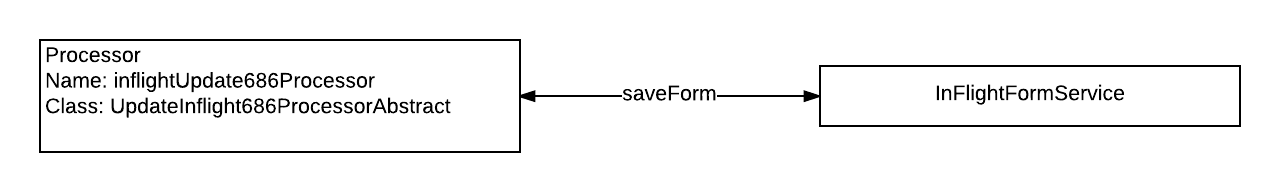
#### inflightProc686Update

This step saves the Form to InFlight.

**Processor**

Name: inflightUpdate686Processor

Class: UpdateInflight686ProcessorAbstract



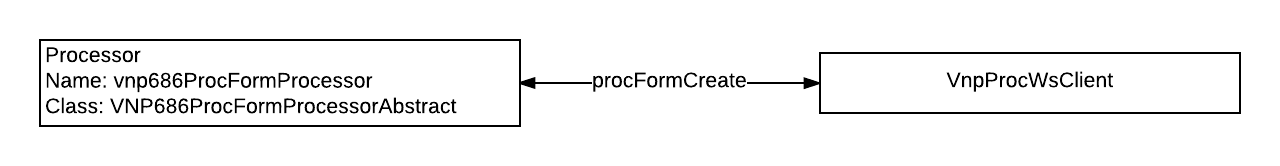
#### vnp686ProcFormCreate

This step creates a new Vnp Process.

**Processor**

Name: vnp686ProcFormProcessor

Class: VNP686ProcFormProcessorAbstract



#### vnp686PtcpntCreate

This step removes the Participant from VNP then recreates it.

**Reader**

Name: vnpPtcpnt686Reader

Class: VNP686PtcpntReader

**Processor**

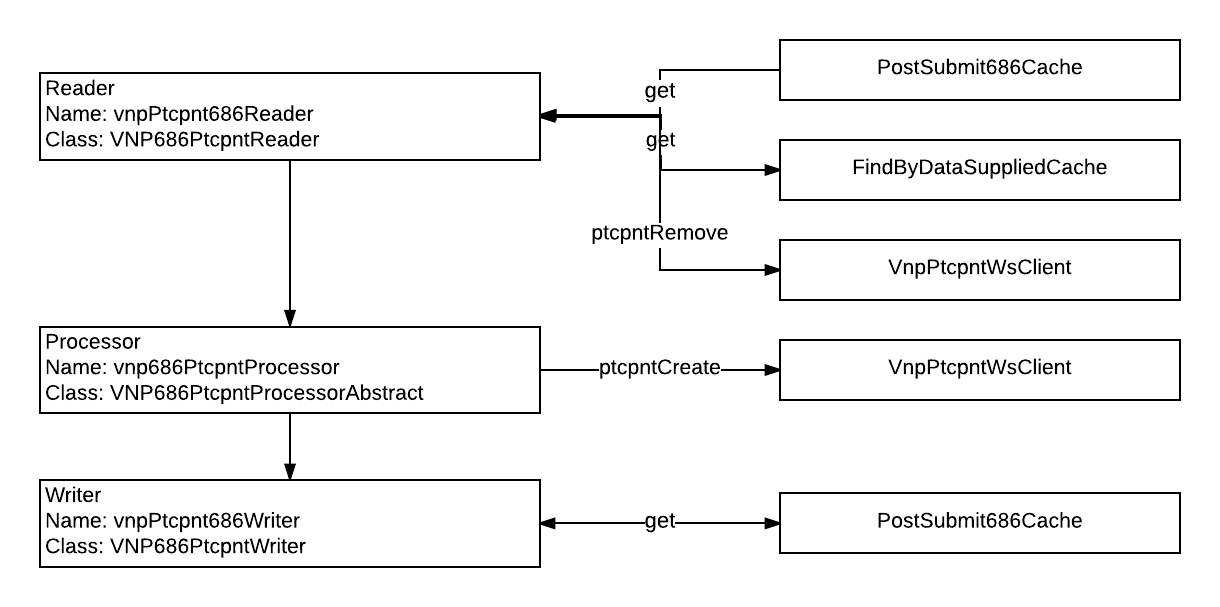
Name: vnp686PtcpntProcessor

Class: VNP686PtcpntProcessorAbstract

**Writer**

Name: vnpPtcpnt686Writer

Class: VNP686PtcpntWriter



#### vnp686PersonCreate

This step removes the Person from VNP then recreates them.

**Reader**

Name: vnpPerson686Reader

Class: VNP686PersonReader

**Processor**

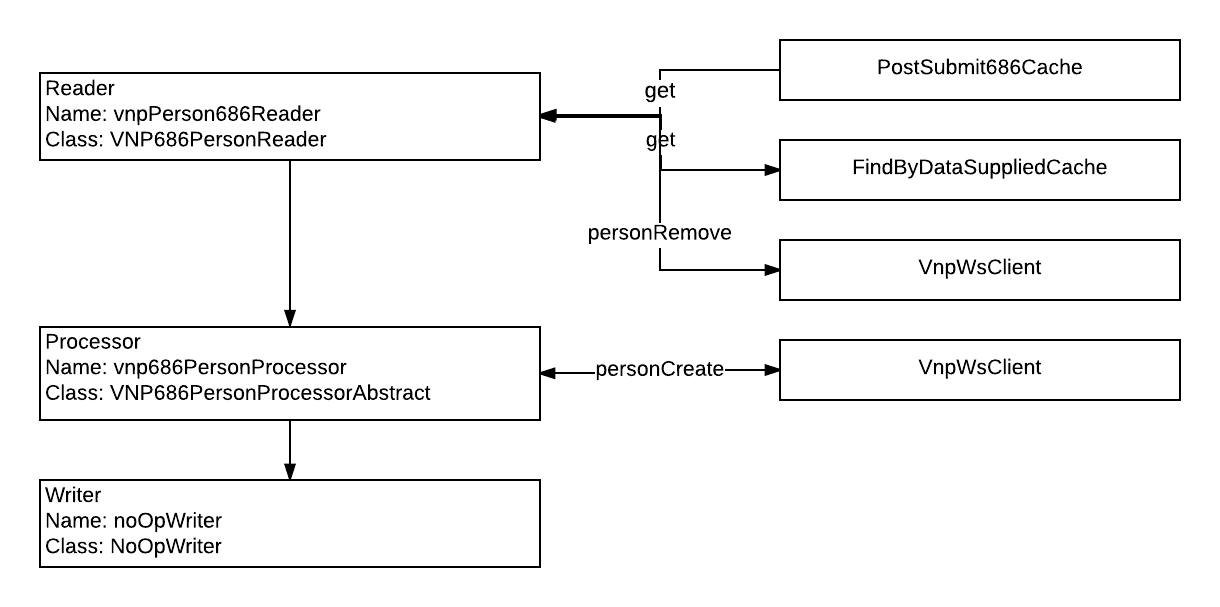
Name: vnp686PersonProcessor

Class: VNP686PersonProcessorAbstract

**Writer**

Name: noOpWriter

Class: NoOpWriter



#### vnp686AddresCreate

This step removes the Address from VNP then recreates it.

**Reader**

Name: vnp686AddressReader

Class: VNP686AddressReader

**Processor**

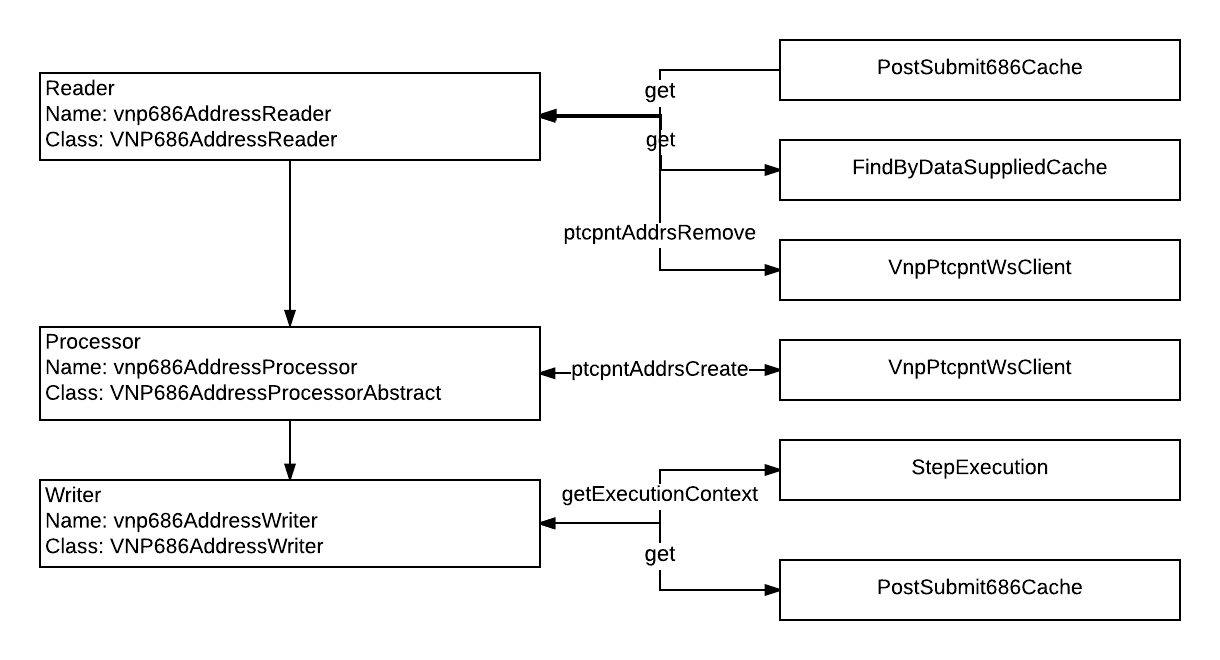
Name: vnp686AddressProcessor

Class: VNP686AddressProcessorAbstract

**Writer**

Name: vnp686AddressWriter

Class: VNP686AddressWriter



#### vnp686PhoneCreate

This step removes the Phone from VNP then recreates it.

**Reader**

Name: vnp686PhoneReader

Class: VNP686PhoneReader

**Processor**

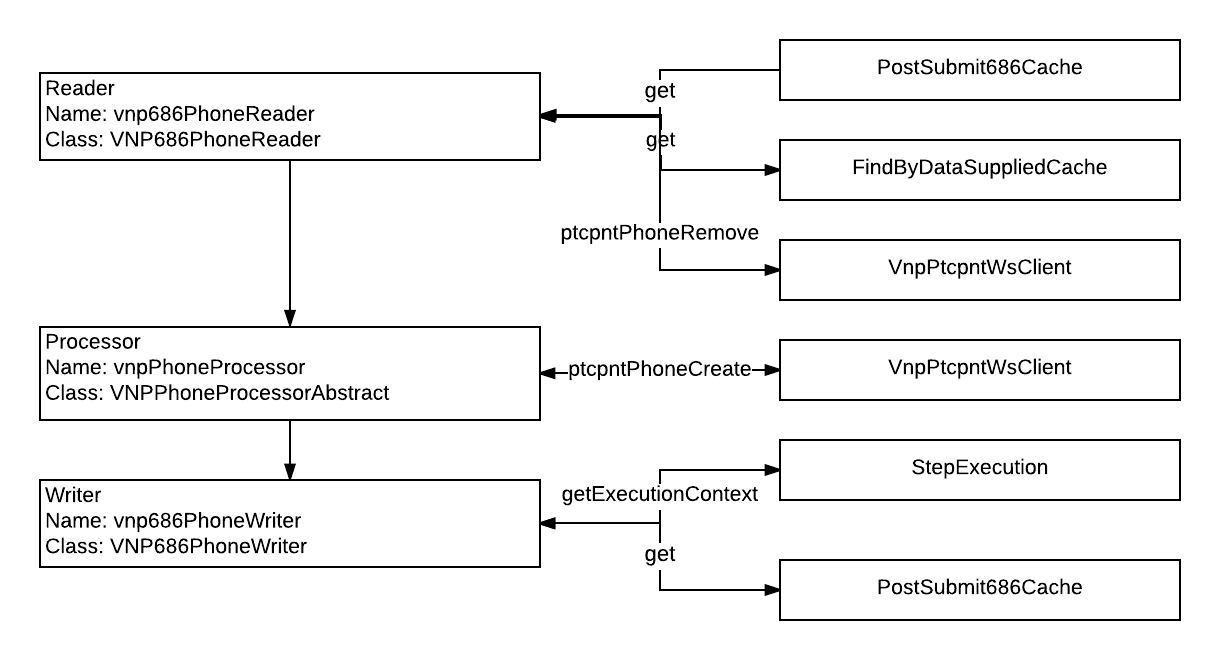
Name: vnpPhoneProcessor

Class: VNPPhoneProcessorAbstract

**Writer**

Name: vnp686PhoneWriter

Class: VNP686PhoneWriter



#### vnp686StudentCreate

This step removes the Student from VNP then recreates them.

**Reader**

Name: vnp686StudentReader

Class: VNP686StudentReader

**Processor**

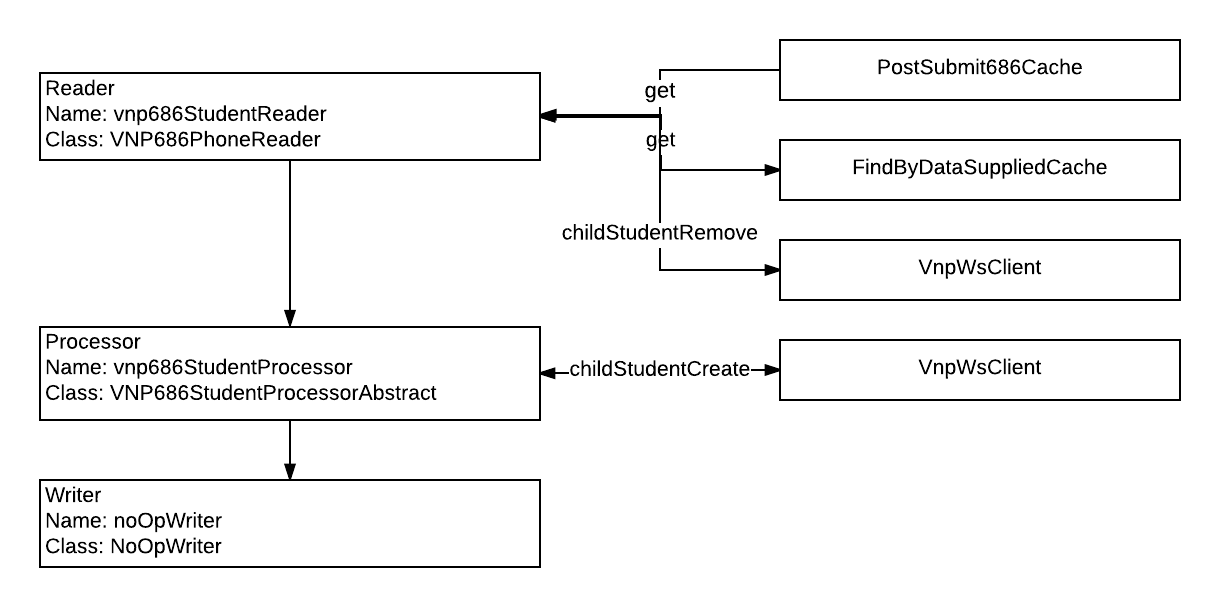
Name: vnp686StudentProcessor

Class: VNP686StudentProcessorAbstract

**Writer**

Name: noOpWriter

Class: NoOpWriter



#### vnp686SchoolCreate

This step removes the School from VNP then recreates it.

**Reader**

Name: vnp686SchoolReader

Class: VNP686SchoolReader

**Processor**

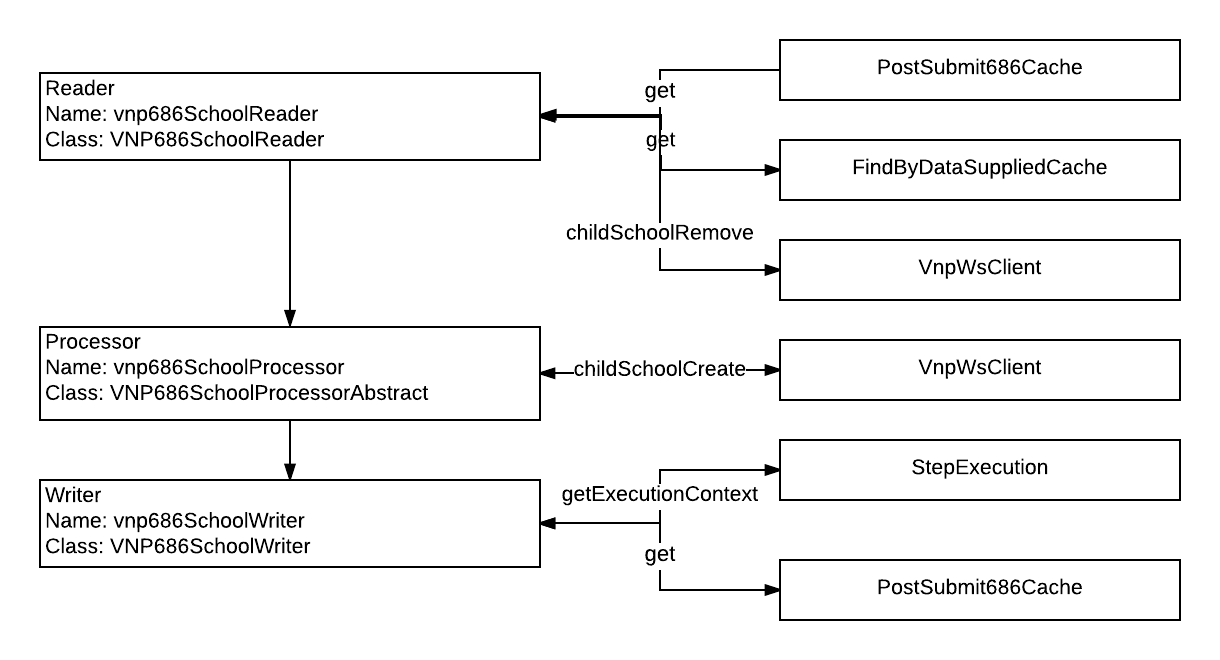
Name: vnp686SchoolProcessor

Class: VNP686SchoolProcessorAbstract

**Writer**

Name: vnp686SchoolWriter

Class: VNP686SchoolWriter



#### vnp686RelationshipCreate

This step removes the Relationship from VNP then recreates it.

**Reader**

Name: vnp686RelationshipReader

Class: VNP686RelationshipReader

**Processor**

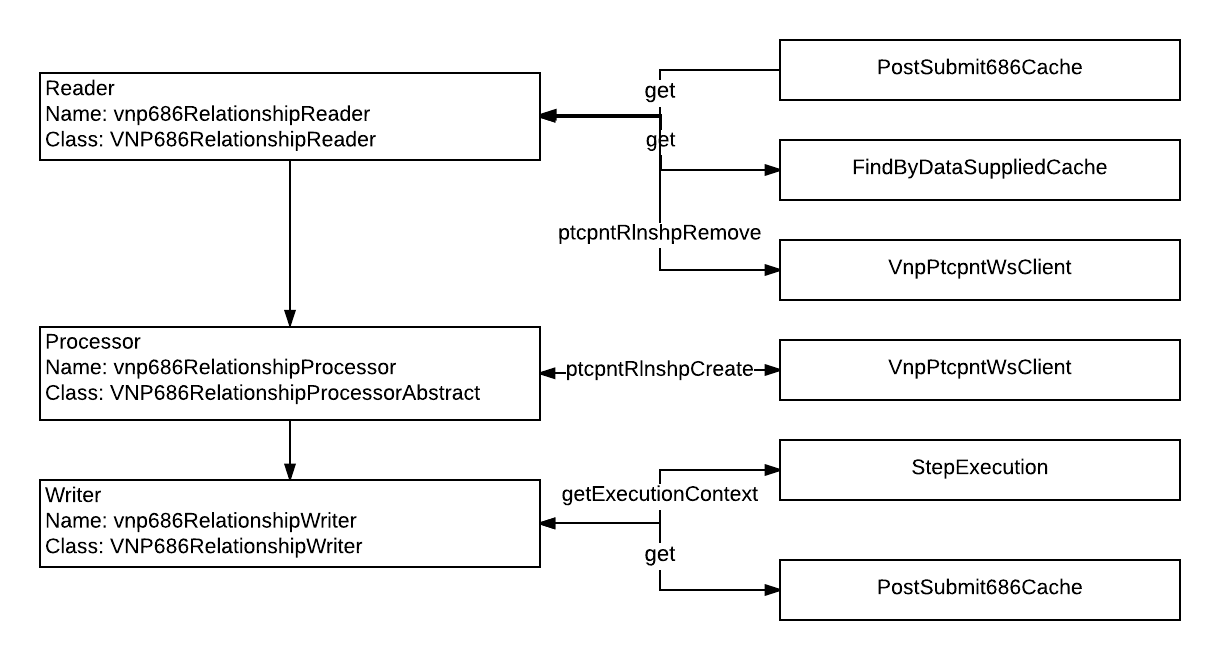
Name: vnp686RelationshipProcessor

Class: VNP686RelationshipProcessorAbstract

**Writer**

Name: vnp686RelationshipWriter

Class: VNP686RelationshipWriter



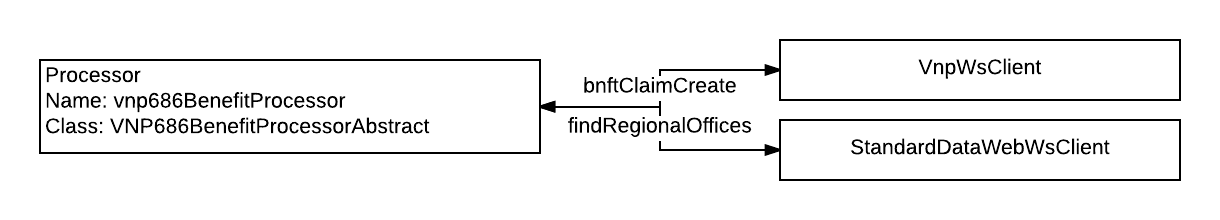
#### vnp686Benefit

This step creates the Benefit Claim in VNP.

**Processor**

Name: vnp686BenefitProcessor

Class: VNP686BenefitProcessorAbstract



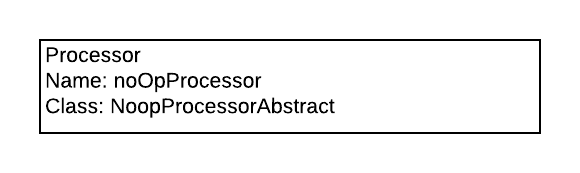
#### vnp686Attachment

This step defines a Processor that does nothing.

**Processor**

Name: noOpProcessor

Class: NoopProcessorAbstract



#### vnp686ProcUpdate

This step updates the Process in VNP.

**Processor**

Name: vnp686ProcUpdateProcessor

Class: VNP686ProcUpdateProcessorAbstract



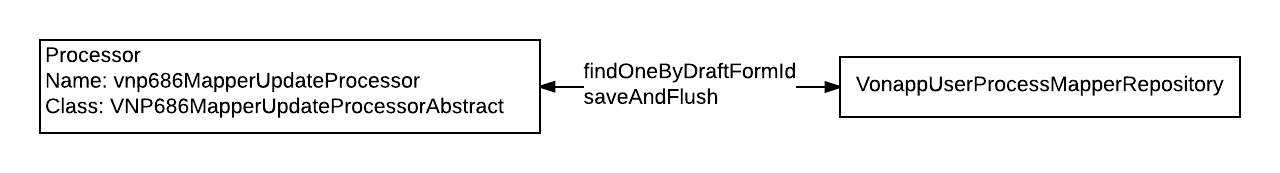
#### vnp686MapperUpdate

This step saves an event to the Vonapp User Process Mapper table.

**Processor**

Name: vnp686MapperUpdateProcessor

Class: VNP686MapperUpdateProcessorAbstract



#### 

#### inflight686Update

This step saves the Form to InFlight and evicts the data from the Caches.

**Processor**

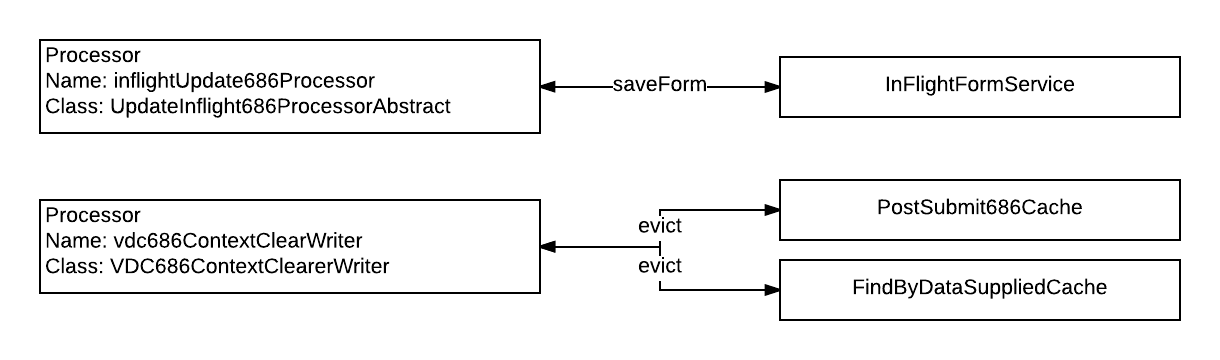
Name: inflightUpdate686Processor

Class: UpdateInflight686ProcessorAbstract

**Writer**

Name: vdc686ContextClearWriter

Class: VDC686ContextClearerWriter



## 

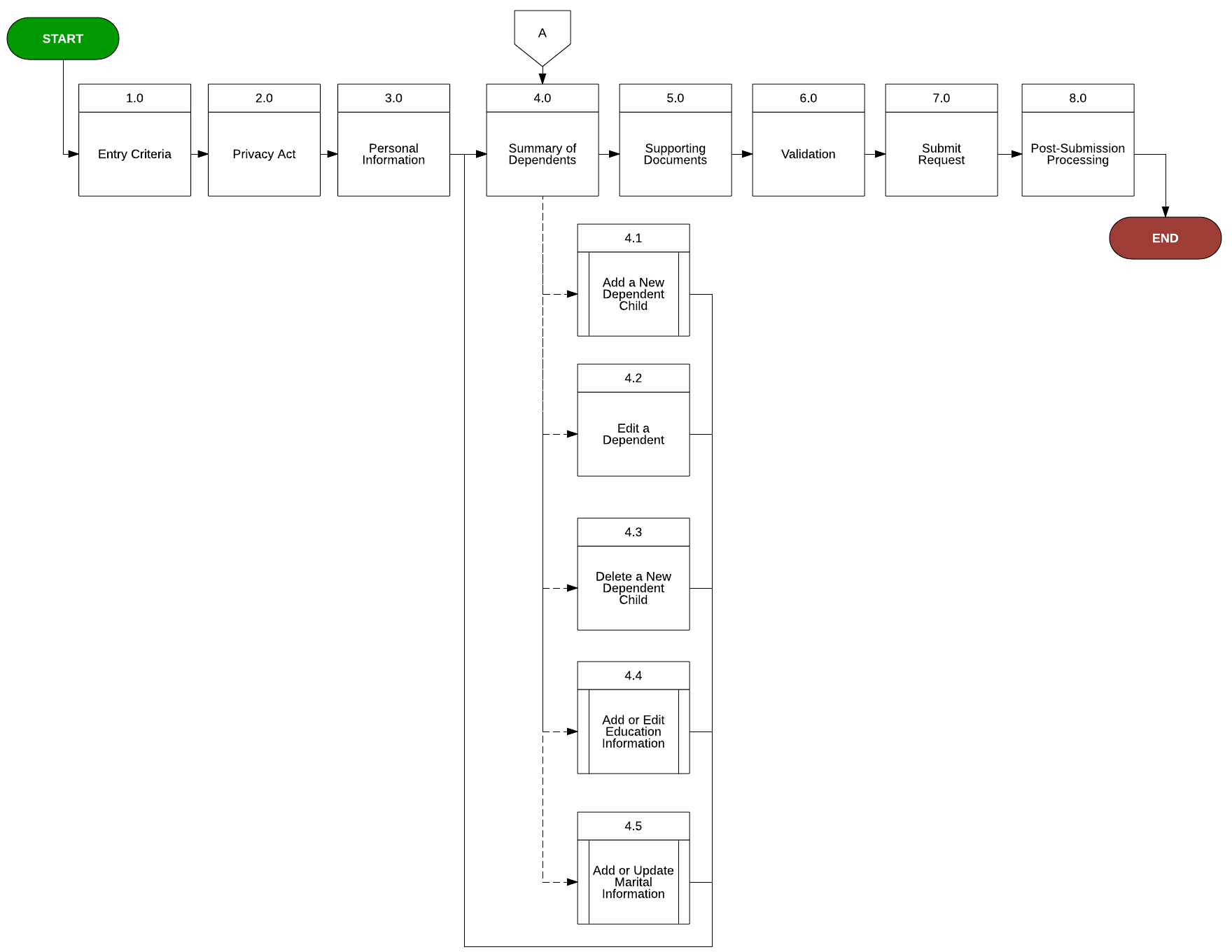
## 

## Use-Cases

<https://csra-evss.atlassian.net/wiki/pages/viewpage.action?pageId=4724876>

### Top Level

This shows in simple flow the path the user takes through the 686/674 application.

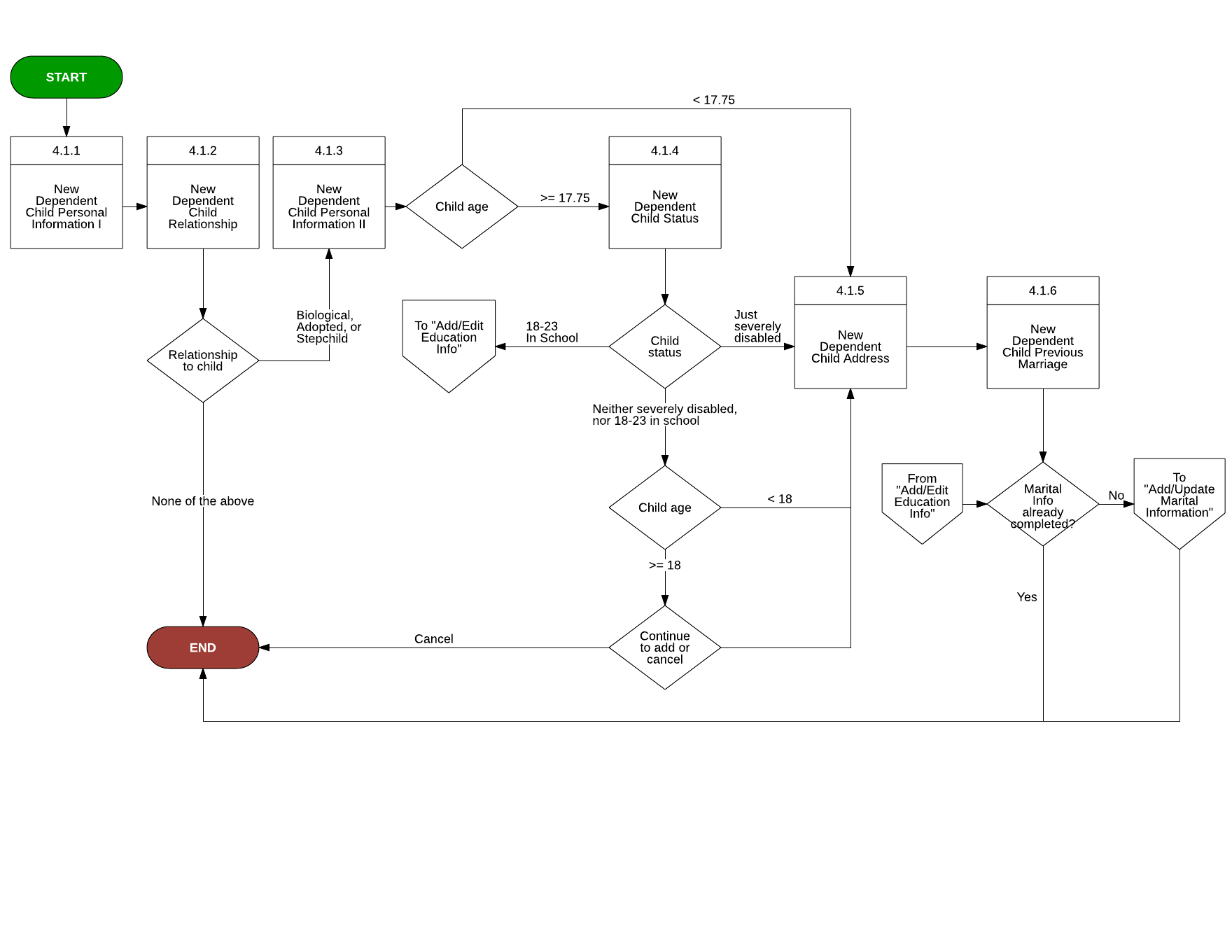


### 

### 

### Add a New Dependent

This is the use-case flow for adding dependents.

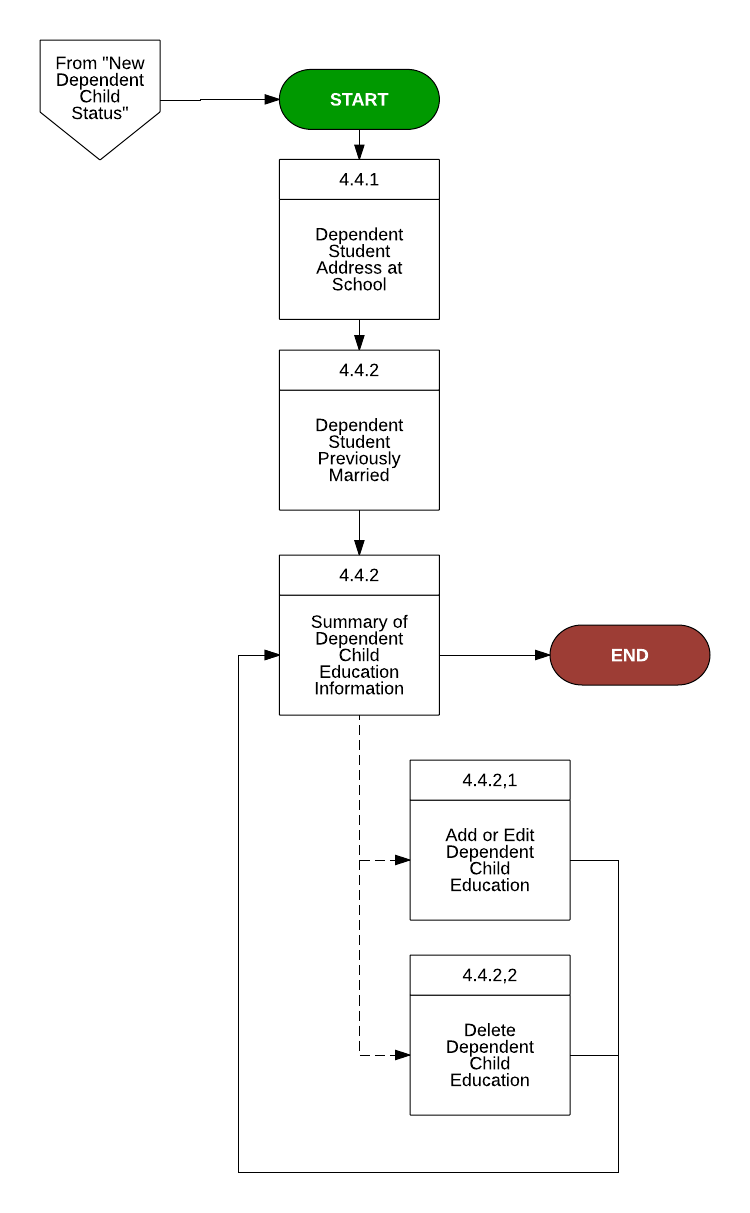


### 

### 

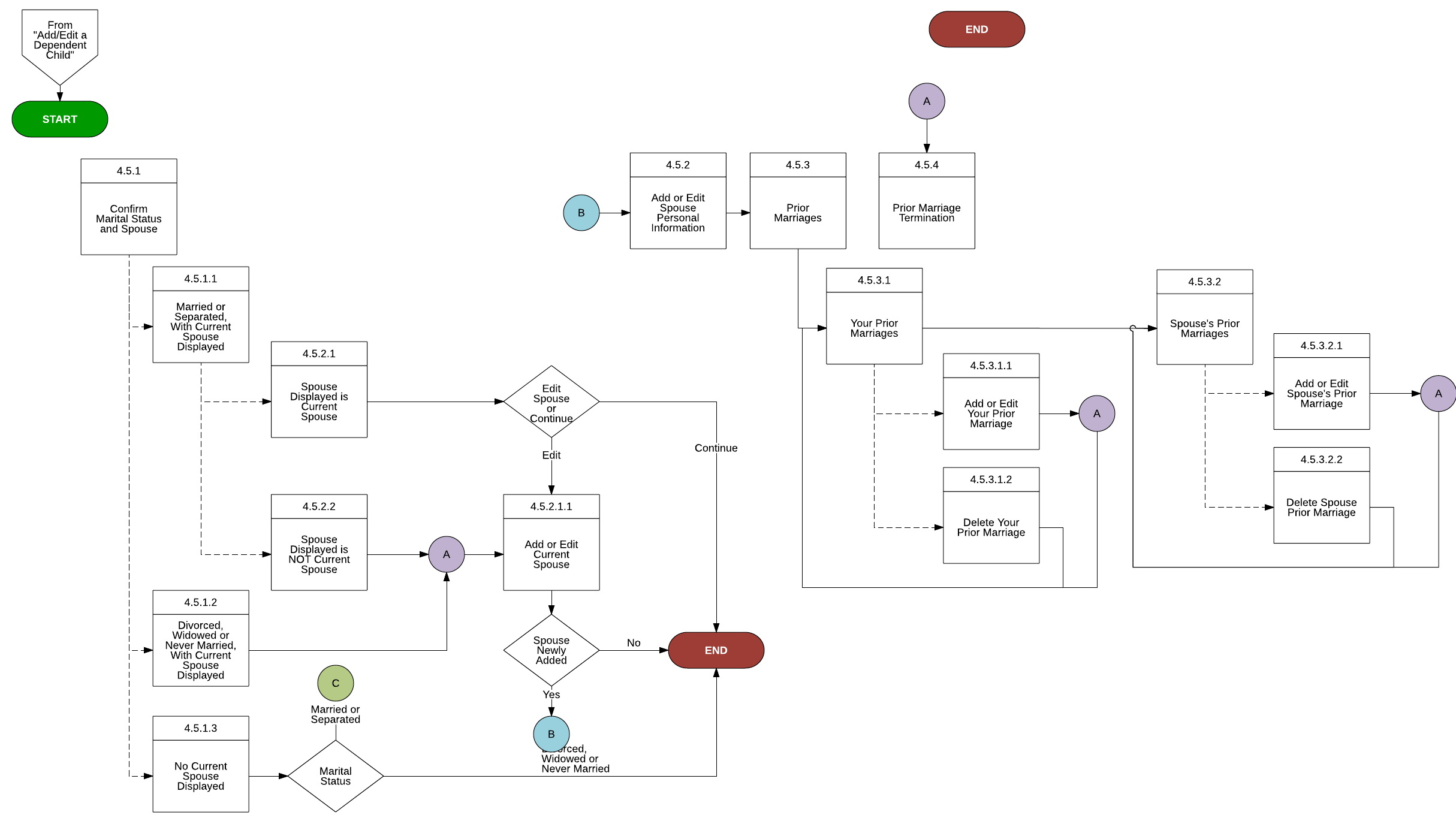
### Add Education

This is the use-case flow for adding education for the veteran, spouse, and children.



### Add Marital Information

This is the use-case flow for adding information about the veteran’s marriage.



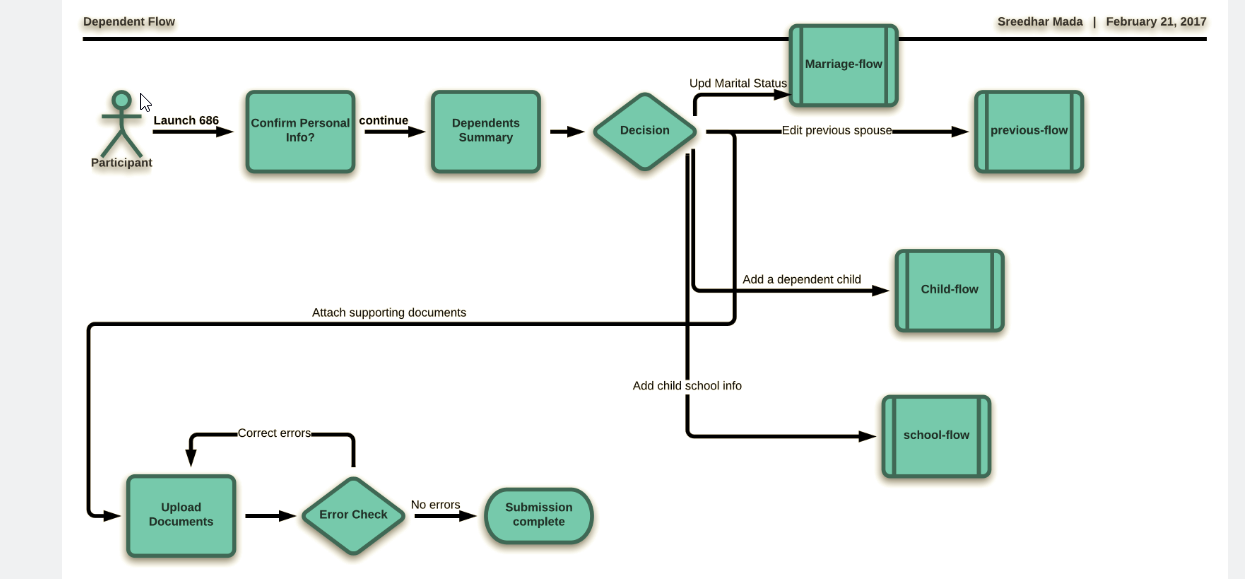
## 

## 

## Web Flow

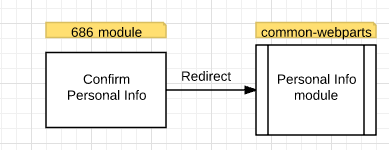
Here is a list of web flows defined in 686.

### Dependent-flow

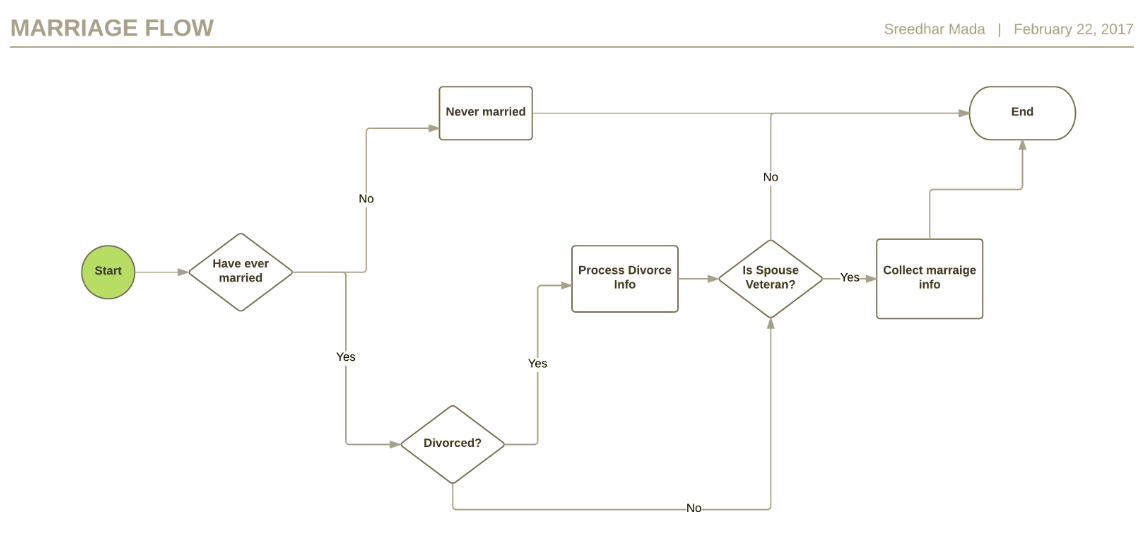


### Service Calls in “Dependent-flow”

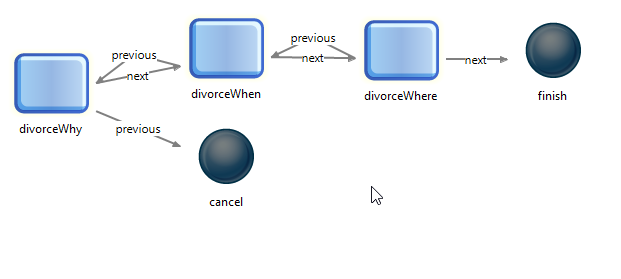
Step



### Marital-flow

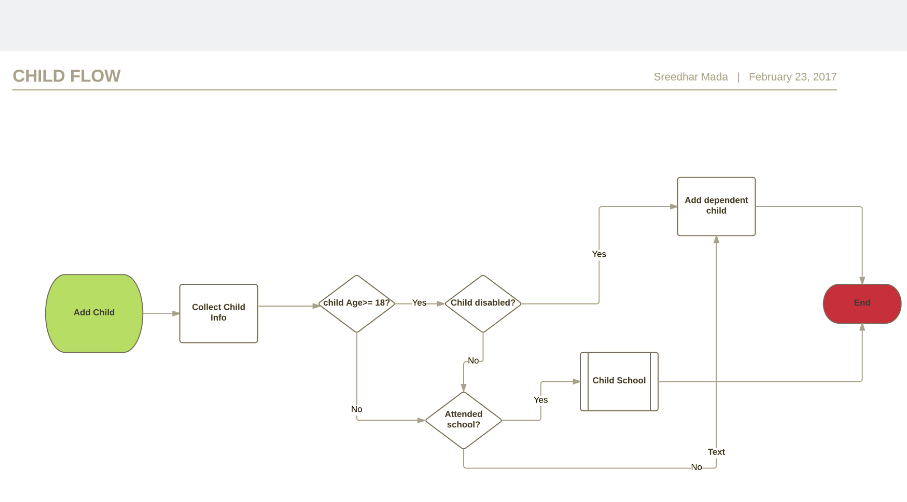


### Divorce-flow

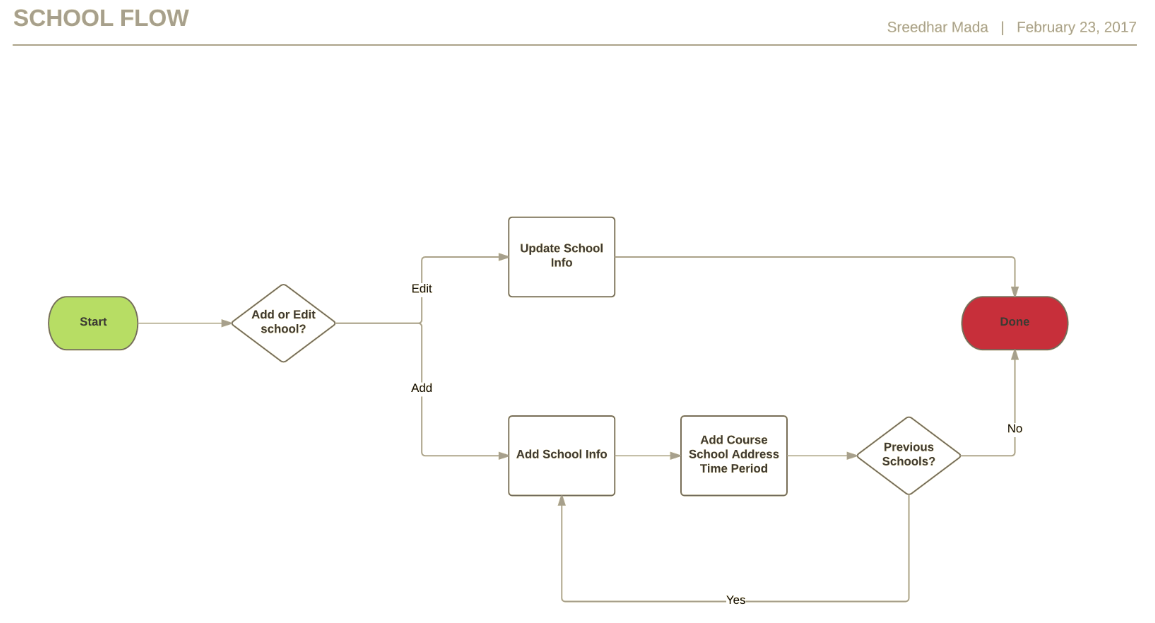


### Previous-flow (For prior marriages)

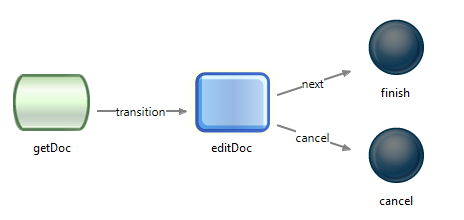
### Child-flow



### School-flow



### Upload-flow



# Appendix

## BGS Services

[https://csra-evss.atlassian.net/wiki/display/VET/BGS+Services](https://csra-evss.atlassian.net/wiki/display/VET/BGS+Services%20)

## System Design Document

[686/674 System Design Document](https://csra-evss.atlassian.net/wiki/pages/viewpage.action?pageId=4465017)

## ICD Document

[686/674 Interface Control Document](https://csra-evss.atlassian.net/wiki/pages/viewpage.action?pageId=4465187)

## Wireframes

Wireframes created for 686 refactor:

[http://g7ky9d.axshare.com](http://g7ky9d.axshare.com/)

[http://sxjm7a.axshare.com](http://sxjm7a.axshare.com/)

Wireframe for 674:

[http://fzp9pm.axshare.com](http://fzp9pm.axshare.com/)

## Test Cases

[686/674 Test Cases](https://csra-evss.atlassian.net/wiki/pages/resumedraft.action?draftId=4464050&draftShareId=35e699a4-ec0c-4f84-ad15-ca4369a33f5d)

# Glossary