

DRAFT

Digits to Digits (D2D)
Electronic Form Ingestion Service

Version 0.5

Interface Control Document



September 26, 2012

DRAFT

Revision History

Date	Version	Description	Author
07/13/2012	0.1	Initial draft	Sonny ElHamahmy
07/29/2012	0.2	Update Business Flow	Sonny ElHamahmy
09/25/2012	0.5	Incorporate VLER Exchange. D2D Section added D2D Documents and Operations described D2D process and context described	Christopher Pryor

Table of Contents

1. Introduction	1
1.1. Purpose	1
1.2. Scope	1
1.3. System Identification	1
1.3.1. Accredited Partner Form Processing Application	1
1.3.2. VLER Gateway	2
1.3.3. D2D Form Ingestion Service	2
2. Interface Definition (VLER)	4
2.1. System Overview	4
2.2. Interface Overview	4
2.2.1. Exchange Protocol	4
2.2.2. Request Web Service	7
2.2.3. Response Web Service	13
2.3. Operations	17
2.4. Security.....	17
3. Interface Definition (D2D Extension)	17
3.1. System Overview	17
3.2. Interface Overview	18
3.2.1. Exchange Protocol	18
3.2.2. Request Documents	18
D2D Document.....	20
D2D Form.....	20
D2D Submission Manifest	20
D2D Submission Confirmation.....	20
VA Forms	21
3.2.3. Response Documents	21
D2D Submission Manifest	21
D2D Submission Confirmation.....	21
3.3. Operations.....	22
3.3.1. Submit Form.....	24
3.3.2. Submit Attachment	24
3.3.3. Check Status	24
3.3.4. Confirm Submission.....	24
Appendix A – Request WSDL Files	25
Appendix B – Response WSDL Files	26

List of Tables

<i>Table 1-1 - Partner Form Processing Application Identifiers</i>	2
<i>Table 1-2 - VLER Gateway Identifiers</i>	2
<i>Table 1-3 - D2D Form Ingestion Service Gateway Identifiers</i>	3
<i>Table 2-1 – Transfer Protocol Parameters</i>	6
<i>Table 2-2 – Request Envelope variable XML Elements</i>	9
<i>Table 3-1 - VLER Operation Name for D2D</i>	18

List of Figures

<i>Figure 2-1 – VLER Gateway Context Diagram</i>	4
<i>Figure 2-2 - Interface extended-Data-Flow-Diagram</i>	5
<i>Figure 2-3 - Web Services</i>	6
<i>Figure 2-4 - Request web service WSDL</i>	8
<i>Figure 2-5 – Example request which should succeed</i>	11
<i>Figure 2-6 – Example web service success return value for a request</i>	12
<i>Figure 2-7 – Example web service failure return value for a request</i>	12
<i>Figure 2-8 – Response web service WSDL</i>	13
<i>Figure 2-9 - Example “success” response</i>	14
<i>Figure 2-10 - Example “failure” response</i>	15
<i>Figure 2-11 - Example web service success return value for a response</i>	16
<i>Figure 2-12 - Example web service failure return value for a response</i>	16
<i>Figure 3-1 - D2D Conceptual Operations</i>	17
<i>Figure 3-2 - D2D Document Structure</i>	19
<i>Figure 3-3 - Submission Overview Sequence</i>	22

1. Introduction

This document is the Interface Control Document (ICD) for the interface between external Accredited Partners (such as Veteran Service Organizations (VSOs)) Electronic Form Processing Applications and the internal D2D Electronic Form Ingestion Service.

The Virtual Lifetime Electronic Record (VLER) Gateway is the exchange through which the two systems shall exchange messages and is the interface that partner applications will consume. This interface is fully described in Section 2.

In this ICD “external” means “external to the VA network” while “internal” means “on the VA network”.

1.1.Purpose

This ICD serves as a specification of the interface between the external VSO applications and the internal D2D Electronic Form Ingestion Service. It will be used by:

1. Developers of external Accredited Partner Electronic Form Processing applications.
2. Developers of the internal D2D Electronic Claims Ingestion application.
3. Developers of the internal VLER Gateway application.

1.2.Scope

This ICD focuses on the software interface between external Accredited Partner Form Processing applications and the internal D2D Electronic Form Ingestion service. It describes the operations (or transaction types), data transfers and communication methods of the service interface supported by the D2D Electronic Form Ingestion application.

Upon formal approval by each participating application, this ICD shall be incorporated into the requirements baseline for each application.

This document does not specify the software design of the VLER Gateway nor of any additional applications which interact with the VLER Gateway.

1.3.System Identification

The systems, applications, and services that this ICD applies to are:

- Accredited Partner Form Processing Application
- VLER Gateway
- D2D Form Ingestion service

1.3.1.Accredited Partner Form Processing Application

To consume the D2D interface an accredited partner must have an electronic system that is used to populate official VA Forms. The system will need to be enhanced to produce and receive the messages as defined in this document.

The following information must be provided by each participating partner to onboard to consume the D2D service though the VLER Gateway. The information will need to be provided for each individual endpoint.

Table 1-1 - Partner Form Processing Application Identifiers

<i>System</i>	<i>Details</i>
Identification number	Unique identifier for the partner's system
Title	VSO Name
Abbreviation	VSO Abbreviation
Vendor	Name of application vendor
Version number	Version of vendor application
Release number	Release number of the vendor application
Point of Contact	
Service Endpoint	URL of the partner application

1.3.2.VLER Gateway

The VLER Gateway serves as a gateway and a processes execution service between the external systems and internal VLER applications. VSOs can electronically submit data and related attachments from their form management systems to the VLER Gateway using a standardized and centralized method .

Table 1-2 identifies the VLER Gateway application for which this ICD defines the communications interface.

Table 1-2 - VLER Gateway Identifiers

<i>System</i>	<i>Details</i>
Identification number	VLER 17
Title	Virtual Lifetime Electronic Record Gateway
Abbreviation	Gateway
Version number	1.0
Release number	1.0
Point of Contact	VA Government Program Manager or Contracting Officer

1.3.3.D2D Form Ingestion Service

The D2D Form Ingestion service is a service that resides on the VA network behind the VLER Gateway exchange. The D2D service handles form submissions and orchestrates submission processing between various VA internal systems as determined by form specific workflows.

Table 1-2Table 1-3 identifies the VLER Gateway application for which this ICD defines the communications interface.

DRAFT

Table 1-3 - D2D Form Ingestion Service Gateway Identifiers

<i>System</i>	<i>Details</i>
Identification number	D2D 1
Title	D2D Form Ingestion Service
Abbreviation	D2D
Version number	1.0
Release number	1.0
Point of Contact	VA Government Program Manager or Contracting Officer

DRAFT

DRAFT

2. Interface Definition (VLER)

This ICD specifies the interface between Accredited Partner Form Submission Applications and the VLER Gateway, as shown in Figure 2-1 – VLER Gateway Context Diagram. The interface is designed to guarantee processing of every request.

Document processing after ingest is handled by the D2D service. Every request is guaranteed to return a success or failure response.

2.1. System Overview

The VLER Gateway serves as a doorway between external partner systems and internal VA applications as shown in Figure 2-2 - Interface extended-Data-Flow-Diagram. The External Application initiates all requests to the VA and will receive a response, for each request, specifying the success or failure of the request.

In this version of the ICD, all requests from the external application are asynchronous requests. Synchronous processing of requests is not supported.

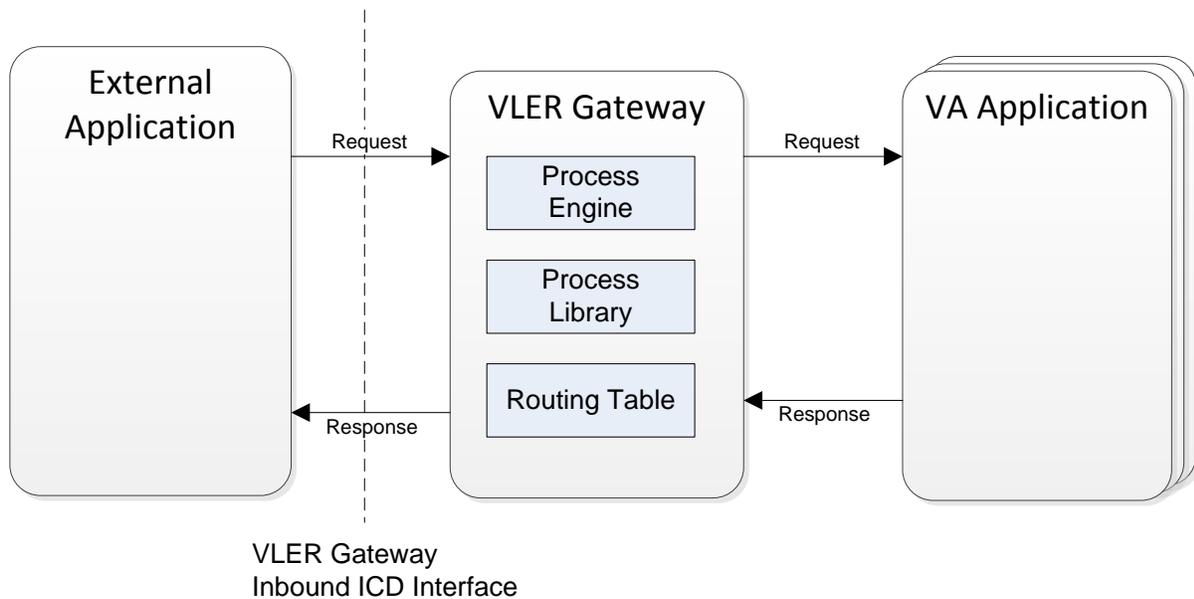


Figure 2-1 – VLER Gateway Context Diagram

2.2. Interface Overview

2.2.1. Exchange Protocol

The communications mechanism selected to operate between the External Applications and the internal VLER Gateway is SOAP based web services as shown in Figure 2-2 - Interface extended-Data-Flow-Diagram and Figure 2-3 - Web Services.

The external application must use a web service to send requests to the VLER Gateway, over both the Internet and the VA Network, while the internal VLER Gateway must use a web service to return responses over the same networks.

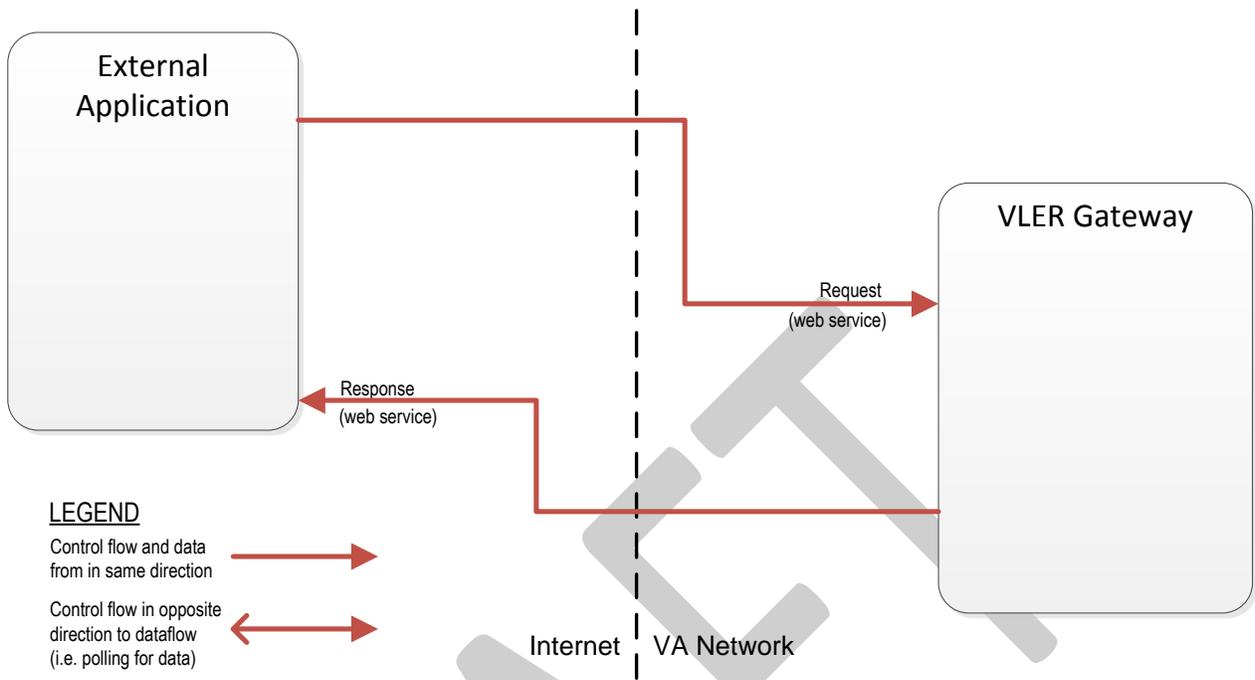


Figure 2-2 - Interface extended-Data-Flow-Diagram

The inbound web service allows the external application to submit requests to VA applications, one at a time, for later, asynchronous, processing by VLER applications.

The request structure is defined in Section 2.2.2 and the response structure in Section 2.2.3. The request and response web services are shown in Figure 3.

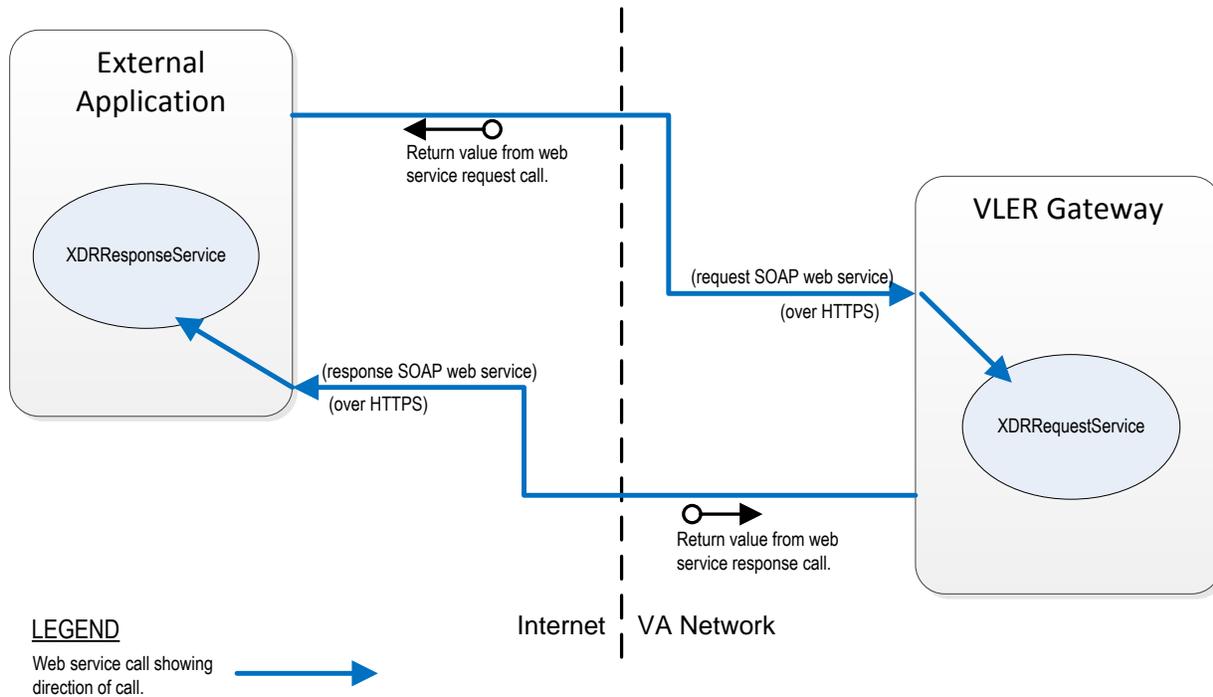


Figure 2-3 - Web Services

This process can only take place when all systems involved in the storage operation are operating in a normal production mode. Planned downtimes by internal VA applications must be allowed for in the external applications' production schedule. See Table 1-2 - VLER Gateway Identifiers . for details of how to obtain the internal systems planned downtimes (e.g. for VLER Gateway and VLER applications).

Table 1-2 - VLER Gateway Identifiers lists parameters of the transfer protocol. These parameters are initial values for some parameters and may change in future versions of this ICD.

Table 2-1 – Transfer Protocol Parameters

Protocol Parameter	Value
VLER Gateway Server URL available on the Internet.	Notional value is VLER.VA.GOV/gateway/ Contact VLER Operations for the production and test values.
External application storage-operation timeout period.	4 hours and should be configurable.
VLER Operations contact.	Contact the VA Government Program Manager or Contracting Officer for details of how to contact operations knowledgeable staff.
Internal systems planned downtimes.	Contact the VA Government Program Manager or Contracting Officer for details of how to obtain scheduled downtime periods.

Protocol Parameter	Value
External application maximum number of transfer retries for any reason.	3
External application maximum request send rate.	25 Hz

The “happy day” sequence of protocol steps (the sending thread) for requests to the VA is as follows:

1. The external application determines the transaction and documents to transfer to the VA.
2. The external application builds the request object and must Base64 encode the wrapped document and insert it into the “Document” element of the request-envelope XML as described in Table 2-2.
3. The external application populates the values of the remaining elements of the request-envelope XML as described in Table 2-2.
4. The external application transfers the request-envelope XML via the request web service call to the VLER Gateway. This web service call constitutes a request to the VA to transmit one document. See Section 2.2.2 for details of the service. There is no guarantee that (1) requests will be processed and stored, even if accepted by VLER Gateway, (2) requests will be processed in the order they are received by the VLER Gateway and (3) a response will be returned for every request.
5. The return-value of the request web service call (returned synchronously to the sender as shown in Figure 2-3 - Web Services) will indicate whether the request was received successfully by the VLER Gateway, for later asynchronous processing and storage by the target VA application.

Some requests from the external application solicit a response from the target VA application. The “happy day” sequence of protocol steps (the acknowledgement thread) to receive back an acknowledgement (the response) for a request is as follows:

6. The external application receives a web service call containing the response to an earlier request.
7. The external application examines the response to determine for which request the response was sent. (See Section 2.2.3 Response Web Service for a description of how to correlate requests with responses.)

In addition to contacting VLER operations for those situations which arise in the above process, VLER Operations must be contacted to notify them of a production problem if:

1. The external application is unable to connect to the VLER Gateway web service for network-related reasons.
2. The VLER Gateway returns error codes for transfers and does not complete inbound or outbound transfers as expected.

See Table 2-1 for details of how to contact VLER Operations.

Section 3 defines the VA application specific details of the ICD.

2.2.2. Request Web Service

The request web service is implemented by the VLER Gateway as shown in Figure 2-4 - Request web service WSDL.

The top level Web Services Description/Definition Language (WSDL) for the service is shown below in **Error! Reference source not found.** The files defining the request web service WSDL are attached in Appendix A – Request WSDL Files.

```
<?xml version="1.0" encoding="utf-8"?>
<definitions
  xmlns:tns="http://_2007.request.async.xdr.iti.ihe/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns:wsaw="http://www.w3.org/2006/05/addressing/wsdl"
  name="XDRRequestService"
  targetNamespace="http://_2007.request.async.xdr.iti.ihe/"
  xmlns="http://schemas.xmlsoap.org/wsdl/">
  <import
    namespace="urn:ihe:iti:xdr:async:request:2007"
    location="XDRRequestService0.wsdl" />
  <types />
  <binding
    xmlns:ns1="urn:ihe:iti:xdr:async:request:2007"
    name="XDRRequest_PortTypeBinding"
    type="ns1:XDRRequest_PortType">
    <wsaw:UsingAddressing />
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http" />
    <operation name="ProvideAndRegisterDocumentSet-bRequest">
      <soap:operation
        soapAction="tns:ProvideAndRegisterDocumentSet-bRequest" />
      <input>
        <soap:body use="literal" />
      </input>
      <output>
        <soap:body use="literal" />
      </output>
    </operation>
  </binding>
  <service name="XDRRequestService">
    <port name="XDRRequest_PortType"
      binding="tns:XDRRequest_PortTypeBinding">
      <soap:address
        location="http://SERVER:PORT/gateway/XDRRequestService" />
    </port>
  </service>
</definitions>
```

Figure 2-4 - Request web service WSDL

DRAFT

The value of the **yellow highlighted** “location” attribute of the “soap:address” element in the WSDL of Figure 2-4 - Request web service WSDL must be specified by VLER Operations in concert with the VLER Gateway team with:

1. “HTTPS” when the communications path is protected by SSL.
2. The fully qualified server URL.
3. The fully qualified VLER Gateway URL path.
4. The port selected for communications with VLER Gateway.

There may be differences between the values for test and production. Contact VLER Operations for the above details (see Table 1-2 - VLER Gateway Identifiers for contacting VLER Operations).

Table 2-2 lists the elements of the request document which may vary from request to request. The value associated with the `<urn3:Slot name="operationName" >` must not be changed as this value indicates the function to be performed by the VLER Gateway. This value is specific to the targeted VA Application and is defined in Section 3.

The instance of the claim form to be stored must be Base64 encoded and added as the value of the “Document” element.

Figure 2-5 shows an example completed envelope document. Requests for test and production use can be created from the XML in Figure 2-5 (used as a template) and modified as described in Table 2-2.

Table 2-2 – Request Envelope variable XML Elements

Element	Description of Element
<code><urn1:SubmitObjectsRequest id=" a498fb06-43b0-452e-94d7-1776d42ce71e "></code>	Required The “id” attribute is the unique identifier of this request. The asynchronous response to this request will have a matching value in the “requestID” attribute of the “RegistryResponse” element. Set the value of this attribute in a request to be the unique identifier of the instance of claim form e.g. a UUID. Type: UUID or unique identifier
<code><urn3:Slot name="operationName" ></code>	Required This slot names the operation to be performed by the VLER Gateway for this request.
<code><urn3:Value>TempOperationName</urn3:Value></code>	Required. The name of the operation to be performed by the VLER Gateway for this request. This value must not be modified and is specific to the targeted VA application (see Table 3-1 - VLER Operation Name for D2D.)

DRAFT

Element	Description of Element
<urn3:Slot name="originatingOrganizationName" >	Required This slot names the originating organization making the request.
<urn3:Value>TempOrganizationName</urn3:Value>	Required Set the value of this slot to be the name of the originating organization making the request. An example value is shown in this ICD. Type: Text
<urn3:Slot name="originatingApplicationName" >	Required This slot names the originating application making the request.
<urn3:Value>TempAppName</urn3:Value>	Required Set the value of this slot to be the name of the originating application making the request. An example value is shown in this ICD. Type: Text
<urn:Document id=" 92004bdd-23a9-4b36-af0b-ee1c830e0cd4">	Required The document "id" attribute is the unique identifier of the instance of the claim form Set the value of this attribute to be a unique identifier of the instance e.g. a UUID. Type: UUID or unique identifier.
<urn:Document >	Required The document element must contain the Base64 encoded value of the instance of an claim form. Type: text (Base64)

DRAFT

Figure 2-5 shows an example request document completed according to the guidelines for elements listed in Table 2-2. The Base64 encoded instance of the internal document (highlighted in green) has been truncated to fit the space available on this page.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:urn="urn:ihe:iti:xds-b:2007"
  xmlns:urn1="urn:oasis:names:tc:ebxml-regrep:xsd:lcm:3.0"
  xmlns:urn2="urn:oasis:names:tc:ebxml-regrep:xsd:rs:3.0"
  xmlns:urn3="urn:oasis:names:tc:ebxml-regrep:xsd:rim:3.0">
  <soapenv:Header/>
  <soapenv:Body>
    <urn:ProvideAndRegisterDocumentSetRequest>
      <urn1:SubmitObjectsRequest id="a498fb06-43b0-452e-94d7-1776d42ce71e">
        <urn2:RequestSlotList>
          <!--Zero or more repetitions:-->
          <urn3:Slot name="operationName" >
            <urn3:ValueList>
              <!--Zero or more repetitions:-->
              <urn3:Value>TempOperationName</urn3:Value>
            </urn3:ValueList>
          </urn3:Slot>
          <urn3:Slot name="originatingOrganizationName" >
            <urn3:ValueList>
              <!--Zero or more repetitions:-->
              <urn3:Value>TempOrganizationName</urn3:Value>
            </urn3:ValueList>
          </urn3:Slot>
          <urn3:Slot name="originatingApplicationName" >
            <urn3:ValueList>
              <!--Zero or more repetitions:-->
              <urn3:Value>TempApplicationName</urn3:Value>
            </urn3:ValueList>
          </urn3:Slot>
        </urn2:RequestSlotList>
      </urn1:SubmitObjectsRequest>
      <!--Zero or more repetitions:-->
      <urn:Document id="92004bdd-23a9-4b36-af0b-ee1c830e0cd4">
        PGdvdI52YS5hZ2VudG9yYW5nZS5mYXN0dHJhY2subW9kZWwuZGJxLmxiay5lYWlyeUNlbGxMZXRr
        ... ..
        ... ..
        aWNhdGlvbN+PC9hbnlPdGh1ckNvbXBsaWNhdGlvbN+DQoJCTwvZ292LnZhLmFnZW50b3Jhbmdl
        LmZhc3R0cmFjay5tb2R1bC5kYnEubHVrLkhhaXJ5Q2VsbExldWt1bWlhPg==
      </urn:Document>
    </urn:ProvideAndRegisterDocumentSetRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

Figure 2-5 – Example request which should succeed

DRAFT

The request web service call from the external application to the VLER Gateway will receive back a return value from the VLER Gateway indicating the success or failure of the submission of a request. If the call is successful then the request will be processed at some future time by VA applications and a response returned. If the call fails then the request must be resubmitted.

Figure 2-6 is an example successful return value. The “message” element contains the return value (highlighted in yellow) and will always start with “SUCCESS”.

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns9:Acknowledgement
      xmlns:ns9="urn:ihe:iti:xdr:2007"
      xmlns:ns8="urn:ihe:iti:xds-b:2007"
      xmlns:ns7="urn:gov:hhs:fha:nhinc:gateway:samltokendata"
      xmlns:ns6="urn:oasis:names:tc:ebxml-regrep:xsd:query:3.0"
      xmlns:ns5="urn:oasis:names:tc:ebxml-regrep:xsd:lcm:3.0"
      xmlns:ns4="urn:oasis:names:tc:ebxml-regrep:xsd:rs:3.0"
      xmlns:ns3="urn:oasis:names:tc:ebxml-regrep:xsd:rim:3.0">
      <message>SUCCESS: Request Received</message>
    </ns9:Acknowledgement>
  </S:Body>
</S:Envelope>
```

Figure 2-6 – Example web service success return value for a request

Figure 2-7 is an example failure return value. The “message” element contains the return value (highlighted in yellow) and will always start with “FAILURE”. Additional text will indicate the nature of the failure.

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns9:Acknowledgement
      xmlns:ns9="urn:ihe:iti:xdr:2007"
      xmlns:ns8="urn:ihe:iti:xds-b:2007"
      xmlns:ns7="urn:gov:hhs:fha:nhinc:gateway:samltokendata"
      xmlns:ns6="urn:oasis:names:tc:ebxml-regrep:xsd:query:3.0"
      xmlns:ns5="urn:oasis:names:tc:ebxml-regrep:xsd:lcm:3.0"
      xmlns:ns4="urn:oasis:names:tc:ebxml-regrep:xsd:rs:3.0"
      xmlns:ns3="urn:oasis:names:tc:ebxml-regrep:xsd:rim:3.0">
      <message>FAILURE:
gov.va.integration.core.component.ComponentProcessorException:
gov.va.integration.core.router.RouterException:
gov.va.integration.core.validator.ValidatorException: Subject Objects Request/Id
cannot be null</message>
    </ns9:Acknowledgement>
  </S:Body>
</S:Envelope>
```

Figure 2-7 – Example web service failure return value for a request

2.2.3. Response Web Service

The response web service must be implemented by the external application as shown in Figure 2-6 – Example web service success return value for a request.

The top level WSDL for the service is shown below in Figure 2-8. The files defining the request web service WSDL are attached in Appendix B – Response WSDL Files

```
<?xml version="1.0" encoding="utf-8"?>
<definitions
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns:tns="http://_2007.response.async.xdr.iti.ihe/"
  name="XDRResponseService"
  targetNamespace="http://_2007.response.async.xdr.iti.ihe/"
  xmlns="http://schemas.xmlsoap.org/wsdl/">
  <import
    namespace="urn:ihe:iti:xdr:async:response:2007"
    location="XDRResponseService0.wsdl" />
  <types />
  <binding
    xmlns:ns1="urn:ihe:iti:xdr:async:response:2007"
    name="XDRResponse_PortTypeBinding"
    type="ns1:XDRResponse_PortType">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http" />
    <operation name="ProvideAndRegisterDocumentSet-bResponse">
      <soap:operation
        soapAction="tns:ProvideAndRegisterDocumentSet-bResponse" />
      <input>
        <soap:body use="literal" />
      </input>
      <output>
        <soap:body use="literal" />
      </output>
    </operation>
  </binding>
  <service name="XDRResponseService">
    <port name="XDRResponse_PortType"
      binding="tns:XDRResponse_PortTypeBinding">
      <soap:address
        location="http://SERVER:PORT/gateway/XDRResponseService" />
    </port>
  </service>
</definitions>
```

Figure 2-8 – Response web service WSDL

The value of the **yellow highlighted** “location” attribute of the “soap:address” element in the WSDL of Figure 2-8 must be specified by the VSO Form Processing application team with:

1. “HTTPS” when the communications path is protected by SSL.
2. The fully qualified external application server URL.
3. The fully qualified external application URL path.
4. The port selected for communications with the external application.

There may be differences between the values for test and production.

The response document (e.g. Figure 2-9), which may be returned to the external application by the VLER Gateway for a request, defines:

1. The original request for which this is the response. A response is the response for a particular request if the value of the **requestID** attribute of the “RegistryResponse” element in the response matches the value of the “id” attribute of the “SubmitObjectsRequest ” element in the associated original request document.
2. The success or failure of the request in the **status** attribute of the “RegistryResponse” element.

Figure 2-9 shows an example successful response document indicating that the request was successfully stored by VA.

```
<env:Envelope xmlns:env="http://schemas.xmlsoap.org/soap/envelope/">
  <env:Header/>
  <env:Body>
    <ns2:RegistryResponse
      xmlns:ns2="urn:oasis:names:tc:ebxml-regrep:xsd:rs:3.0"
      xmlns="urn:oasis:names:tc:ebxml-regrep:xsd:rim:3.0"
      xmlns:ns4="urn:oasis:names:tc:ebxml-regrep:xsd:lcm:3.0"
      xmlns:ns3="urn:oasis:names:tc:ebxml-regrep:xsd:query:3.0"
      requestId="a498fb06-43b0-452e-94d7-1776d42ce71e"
      status="urn:oasis:names:tc:ebxml-regrep:ResponseStatusType:Success"/>
  </env:Body>
</env:Envelope>
```

Figure 2-9 - Example “success” response

DRAFT

Figure 2-10 shows an example failure response document indicating that the request was not successfully stored by VA.

```
<env:Envelope xmlns:env="http://schemas.xmlsoap.org/soap/envelope/">
  <env:Header/>
  <env:Body>
    <ns2:RegistryResponse
      xmlns:ns2="urn:oasis:names:tc:ebxml-regrep:xsd:rs:3.0"
      xmlns="urn:oasis:names:tc:ebxml-regrep:xsd:rim:3.0"
      xmlns:ns4="urn:oasis:names:tc:ebxml-regrep:xsd:lcm:3.0"
      xmlns:ns3="urn:oasis:names:tc:ebxml-regrep:xsd:query:3.0"
      requestId="63101add-a249-49d6-a6bc-026e55a49283"
      status="urn:oasis:names:tc:ebxml-regrep:ResponseStatusType:Failure">
      <ns2:RegistryErrorList>
        <ns2:RegistryError
          codeContext="Something went wrong with the application"
          errorCode="XDSRegistryMetadataError"
          severity="urn:oasis:names:tc:ebxml-
            regrep:ErrorSeverityType:Error"/>
      </ns2:RegistryErrorList>
    </ns2:RegistryResponse>
  </env:Body>
</env:Envelope>
```

Figure 2-10 - Example “failure” response

The “RegistryErrorList” element provides additional information in the response about the error which caused the failure of the request.

DRAFT

The response web service call from the VLER Gateway to external application must receive back a return value from the external application indicating the success or failure of the receipt of the response by the external application.

The processing carried out by the external application on receipt of the response from the VLER Gateway is outside the scope of this ICD.

Figure 2-11 is an example successful return value. The “message” element contains the return value and will always start with “SUCCESS”.

```
<soapenv:Envelope
  xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:urn="urn:ihe:iti:xdr:2007">
  <soapenv:Header/>
  <soapenv:Body>
    <urn:Acknowledgement>
      <message>SUCCESS</message>
    </urn:Acknowledgement>
  </soapenv:Body>
</soapenv:Envelope>
```

Figure 2-11 - Example web service success return value for a response

Figure 2-12 is an example failure return value. The “message” element contains the return value and will always start with “FAILURE”. Additional text will indicate the nature of the failure.

```
<soapenv:Envelope
  xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:urn="urn:ihe:iti:xdr:2007">
  <soapenv:Header/>
  <soapenv:Body>
    <urn:Acknowledgement>
      <!--Optional:-->
      <message>FAILURE: Something went wrong</message>
    </urn:Acknowledgement>
  </soapenv:Body>
</soapenv:Envelope>
```

Figure 2-12 - Example web service failure return value for a response

2.3. Operations

This interface supports one inbound operation, namely a request containing a document, and one outbound operation, namely a response containing the success or failure of a request.

The request operation is invoked by the external application calling a web service on the VLER Gateway.

The response to the request is invoked by the VLER Gateway calling a web service on the external application.

2.4. Security

Transfers are protected by HTTPS using certificates which identify computers external application and the VLER Gateway applications. The certificates for the external application and the VLER Gateway computers must be securely exchanged. See Table 1-2 - VLER Gateway Identifiers for contact information.

3. Interface Definition (D2D Extension)

The VLER Interface described in Section 2 is extended to support the D2D Application as is described in this section.

3.1. System Overview

D2D enables Accredited Partners, such as a Veteran Service Organizations (VSO,) to digitally submit VA Forms and attachments to the VA. D2D will eventually support a large number of forms that have unique structure and content, may be accompanied by a substantial number of supporting documentation, and are processed along form-type-specific workflows. The pilot form for the D2D exchange is the 21-526E.

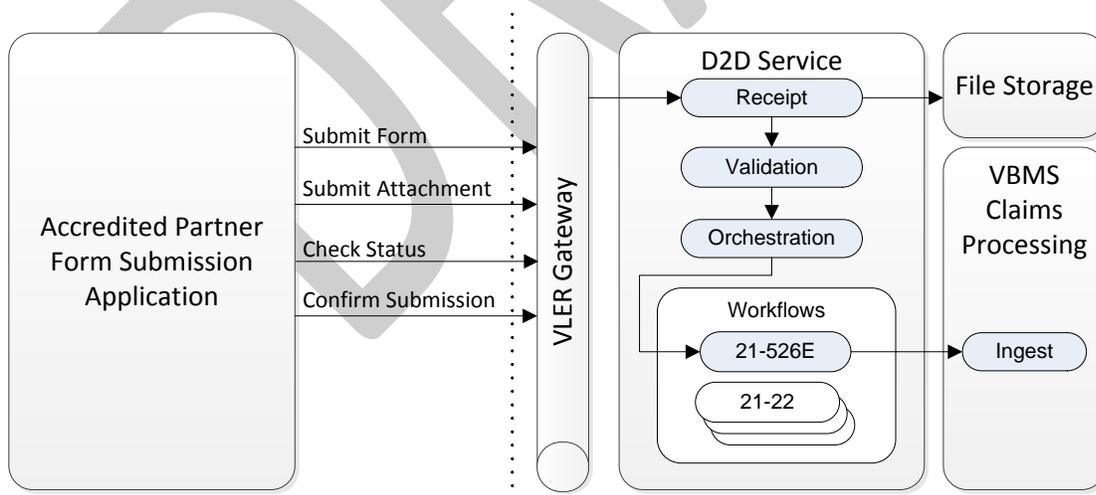


Figure 3-1 - D2D Conceptual Operations

The D2D document exchange supports four documents that correspond to conceptual operations;

1. Submit Form

2. Submit Attachment
3. Confirm Submission
4. Check Status

3.2.Interface Overview

The value of the VLER operationName from Table 2-2 – Request Envelope variable XML Elements in for D2D is;

Table 3-1 - VLER Operation Name for D2D

Element	Description of Element
<urn3:Value> D2D.SubmitDocument </urn3:Value>	<p>Required. The name of the operation to be performed by the VLER Gateway for this request.</p> <p>This value must not be modified and is specific to the targeted VA application (see Section 3.)</p>

The D2D exchange enables the transfer of electronic documents in an asynchronous manner. The size and number of supporting documents that may accompany a single form precludes a synchronous solution.

A Submission is for a single Veteran and must contain at least one VA Form and may contain any number of attachment documents. Individual document size is restricted by internal storage constraints and extremely large documents may need to be partitioned into multiple files.

Each document in a submission is sent with a unique Transaction ID that identifies the document, and there is no implied or required order of transmission and receipt, all individual document transfers are stateless transfers. Documents are linked together because they contain the same Submission ID.

Partner applications can request a Submission Manifest from the D2D Service that details the known composition and state of the Submission and Partner applications can send a Submission Confirmation document to request the D2D Service to request the service to begin processing the Submission into VA internal systems (for instance; to establish a claim.)

See Section 3.2.2 Request Documents for a complete description of D2D documents.

3.2.1.Exchange Protocol

The Exchange Protocol for D2D is fully described in Section 2.2.1. Partner applications will transfer files to D2D operations using the described VLER Request and Response web services.

3.2.2.Request Documents

For each request, the Partner application generates a document that is base64 encoded in the VLER ProvideAndRegisterDocumentSetRequest object. This D2D Document contains metadata about a wrapped document that is base64 encoded in the D2D Document. If the document is a D2D Form, then it

DRAFT

will contain a third layer document that contains the populated xml of the specified form as illustrated in Figure 3-2 - D2D Document Structure.

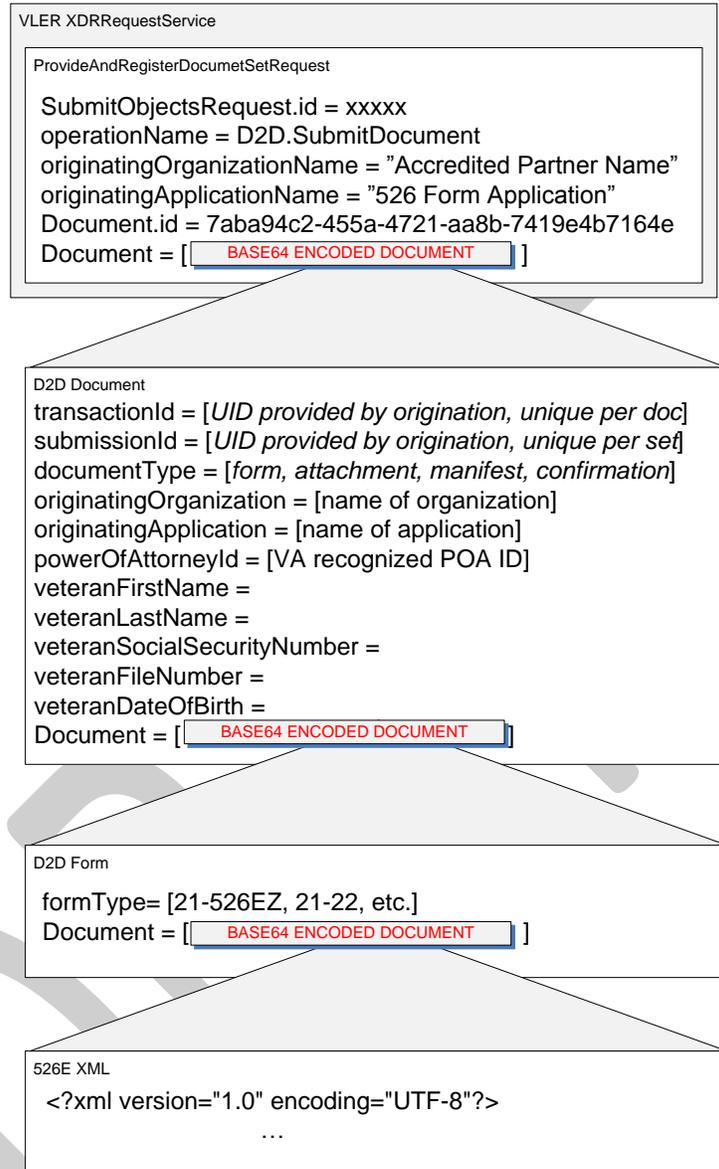


Figure 3-2 - D2D Document Structure

These documents are xml structures that will be formally defined with XML schemas in published XSD files. These are currently under construction.

DRAFT

D2D Document

The root D2D Document envelope contains an embedded base64 encoded document and metadata that:

1. Identifies the Veteran
2. Identifies the Accredited Partner
3. Identifies the Transmission
4. Describes the Encoded Document

The D2D Service uses the metadata contained in the D2D Document to;

1. Validate the Veteran
2. Validate the POA relationship of the Accredited Partner to the Veteran
3. Determine the workflow and processing rules for the document.

The embedded documents will be;

1. A D2D Form,
2. An attachment (PDF or other non-interpretable format),
3. A D2D Submission Manifest,
4. Or a D2D Submission Confirmation

D2D Form

The D2D Form envelope contains metadata that describes the form (such as Form Name; 21-526E, 21-22, etc. and format version) and an embedded base64 encoded VA Form document that is XML conforming to the defined data structure of that form.

D2D Submission Manifest

The D2D Submission Manifest contains metadata that describes a submission and metadata that describes the contents of the submission such as;

1. Status of the Submission
2. All Transaction Ids
3. Status of documents (known, unknown, saved, transaction IDs)
4. Date of receipt(s)
5. Processing state for documents or forms (processed, unprocessed)
6. Errors, issues or response messages related to the submission or particular documents.

The detail and structure of the Submission Manifest provides a current and complete assessment of the submission as the D2D Service has. The visibility into the state within the VA enables the Partner Application respond to certain information (that D2D does not recognize a document that the partner expects to be contained in the submission so the partner application can resend) or save the information for manual correction (if a Veteran is not recognized or a POA is not in place.)

D2D Submission Confirmation

The D2D Submission Confirmation is the mechanism for the Partner Application to instruct the D2D Service to proceed with processing the submission, or parts of the submission, into other VA systems. It contains;

1. Submission Id
2. Target Document(s)
3. Action(s)

VA Forms

The pilot submission for the D2D implements the 21-526EZ as the 526E. The structure of this document will be coordinated with internal VA Claims Processing systems such as VBMS and VDC.

Each VA Form will have a XML Schema Definition (XSD) that specified the formal structured data representation of the form. Each XSD will be versioned, controlled and officially published as part of the D2D specification as capabilities evolve.

3.2.3. Response Documents

These documents are xml structures that will be formally defined with XML schemas in published XSD files. These are currently under construction.

D2D Submission Manifest

The response Submission Manifest will have the same structure and content as the request manifest document but will be augmented with timestamp, status and error or response values and codes from the D2D system.

D2D Submission Confirmation

The response Submission Confirmation will have the same structure as the corresponding request document but will be augmented with timestamp, actions taken, results of actions and any error or response values and codes from VA internal processing.

3.3.Operations

The four conceptual operations are used together to submit forms to the VA as illustrated by Figure 3-3 - Submission Overview Sequence, a sample “happy path” submission with a single attachment.

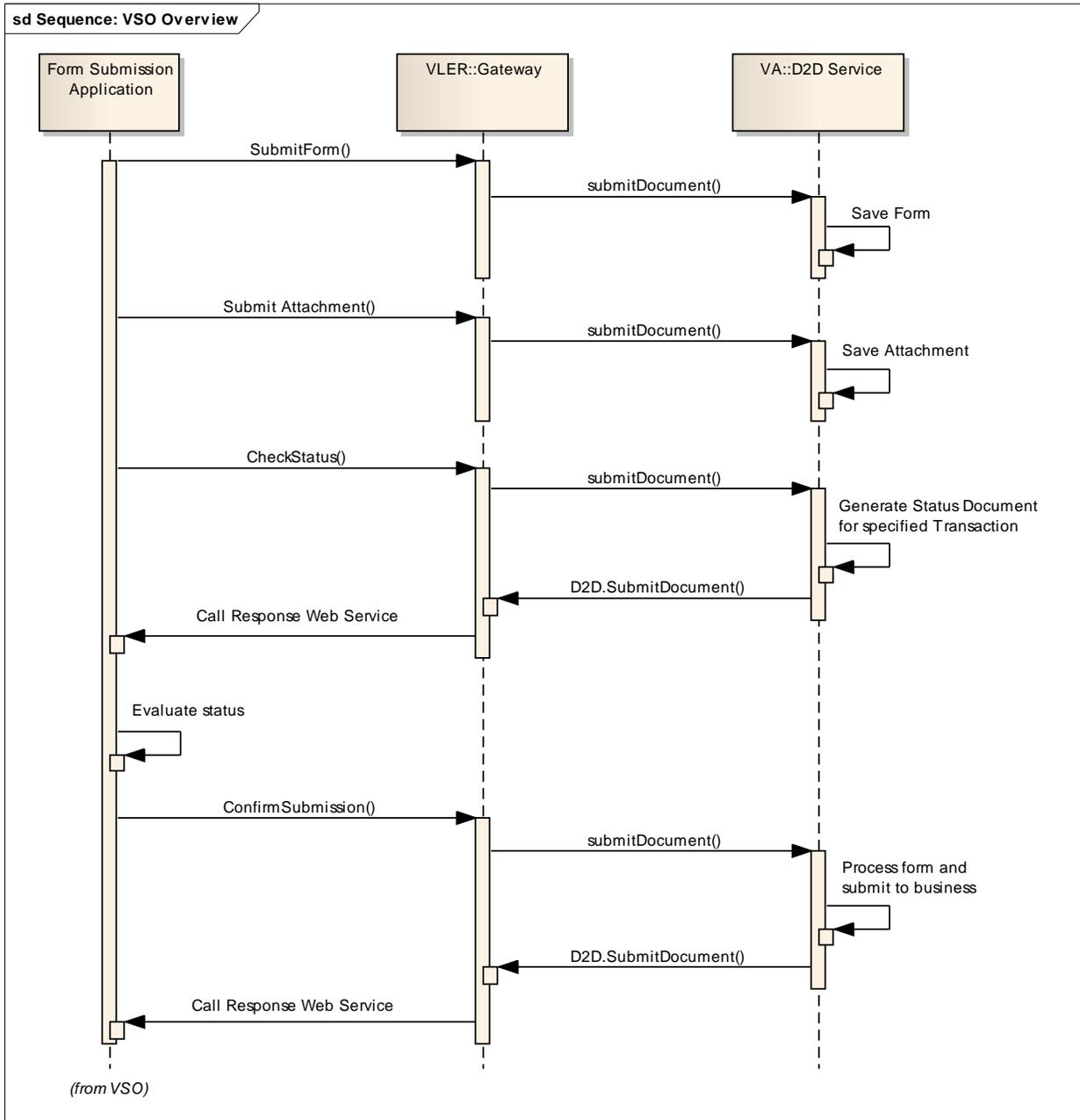


Figure 3-3 - Submission Overview Sequence

DRAFT

The sequence of events in Figure 3-3 - Submission Overview Sequence is as follows;

Submit Form:

1. The partner application is used to prepare the claim and assemble all electronic documents.
2. The partner application generates a unique Submission ID that is used to link separate documents to the same submission.
3. The partner application generates the 526E xml document, wraps it with D2D Form, D2D Document and VLER Request envelopes and calls the VLER XDRRequestService.
4. The VLER Gateway transfers the document to the D2D Service
5. The D2D Service receives the document, determines it is a form, decodes the form, validates the schema, timestamps the receipt, saves the document to the internal repository and registers the submission ID.

Submit Attachment:

1. The partner application wraps each attachment with the VLER Request Envelope and calls the VLER XDRRequestService. Each request contains the same submission ID that links the document to other documents.
2. The VLER Gateway transfers the document to the D2D Service
3. The D2D Service receives the document, determines it is an attachment, saves the document to the internal repository and registers the document with the submission ID.
4. The partner application repeats this process for each attachment. A submission may contain any number of forms and attachments and the D2D Service does not require a particular order of receipt.

Check Status:

1. When the Partner Application is finished sending the documents in the submission to the D2D service it generates a Submission Manifest document and sends this to the D2D Service in order to Check Status.
2. In response to the Submission Manifest, the D2D Service generates another Submission Manifest that includes the status of each form and attachment and sends this, through VLER, to the XDRResponseService at the endpoint specified by the partner application.
3. The manifest lists each specified document, its status (success, error, not found,) and any errors or validation issues encountered.
4. The Partner Application evaluates the manifest to determine if the Submission was successfully completed and if the current state of the Submission in D2D is ready for processing. The details in the manifest enable the Partner Application to determine any required corrective actions.

Confirm Submission:

1. When the Partner Application determines that the Submission is complete and is ready for the D2D Service to process the Submission on to business, the partner application sends a Submission Confirmation document.
2. When the D2D Service receives a Submission Confirmation document, it applies the appropriate workflow to the Submission and starts the orchestration to pass the data, forms and attachments on to VA business applications.
3. Business specific values (such as Claim ID) and errors or issues are collected in a Submission Confirmation document that is sent to the Partner Application.
4. After confirmation, the submission is no longer an active submission and is removed from the D2D registry.

DRAFT

3.3.1. Submit Form

The Submit Form operation occurs when the Partner Application sends a D2D Form in the D2D Document. D2D saves the form as a document in the document store. Validates the form against the form schema, and performs form specific validation including communication with other VA systems. The interaction with certain VA systems crosses over from technical to legal and policy domains and this document does not address any of those issues. Each form workflow will separately address the business process pertaining to the form and the policy and legal impact of the workflow.

Each D2D Submission must contain at least one form. Some submissions will contain multiple forms (such as a 21-22 Power of Attorney form accompanying a 21-526 Claims form.) The pilot for D2D will only feature the 21-526E but additional capabilities are already being considered.

3.3.2. Submit Attachment

The Submit Attachment operation occurs when the Partner Application sends a document that is not a VA Form. D2D saves the document in the document store and performs security checks on the file, but no additional processing is indicated for an attachment. A Submission may have any number of attachments within to-be-established limits.

3.3.3. Check Status

A Check Status operation occurs when a Partner Application sends a D2D Submission Manifest. The D2D service returns to the Partner Application all available status information on the indicated submission. The Check Status operation does not change the state of the submission, it is a read-only/report activity. For security and privacy reasons, only the originating Partner can check the status of a submission and certain information may be restricted to an established POA.

The goal is to provide Partner Applications as much information as possible to enable automation, workflow integration and flexible synchronization of business processes. This capability is focused on the status of the submission itself, not necessarily the status of the business process that will be performed once the submission is complete (for instance, the status of a Claim.)

3.3.4. Confirm Submission

A Confirm Submission operation is performed when a Partner Application sends a D2D Submission Confirmation document. The D2D Service reads the confirmation and attempts to perform the indicated actions for the indicated documents or entire submission. The D2D Service orchestrates the submission to completion and sends a D2D Submission Confirmation to the originating Partner Application that contains status, results, response messages and any error messages encountered during processing. Confirm Submission is the final stage of the D2D Service and ends the lifespan of the Submission. If a Submission in a known error state is Confirmed, it will be deleted without processing.

Appendix A – Request WSDL Files

The WSDL files for the request service to be implemented by the VLER Gateway are included below in this ICD.

Embedded File	Embedded File	Description
 C:\XDR\ XDRRequestService\		Top level WSDL file.
 C:\XDR\ XDRRequestService\		Definitions file.
 C:\XDR\ XDRRequestService\	 C:\XDR\ XDRRequestService\	XSD
 C:\XDR\ XDRRequestService\	 C:\XDR\ XDRRequestService\	XSD
 C:\XDR\ XDRRequestService\	 C:\XDR\ XDRRequestService\	XSD
 C:\XDR\ XDRRequestService\		XSD
 C:\XDR\ XDRRequestService\		XSD
 C:\XDR\ XDRRequestService\		XSD

Appendix B – Response WSDL Files

The WSDL files for the response service to be implemented by the VSO Form Processing application are included below in this ICD.

Embedded File	Embedded File	Description
 C:\XDR\ XDRResponseService		Top level WSDL file.
 C:\XDR\ XDRResponseService		Definitions file.
 C:\XDR\ XDRResponseService	 C:\XDR\ XDRResponseService	XSD
 C:\XDR\ XDRResponseService	 C:\XDR\ XDRResponseService	XSD
 C:\XDR\ XDRResponseService	 C:\XDR\ XDRResponseService	XSD
 C:\XDR\ XDRResponseService		XSD
 C:\XDR\ XDRResponseService		XSD
 C:\XDR\ XDRResponseService		XSD