Cascading Data Protocol Handlers - Working Example with TCP Virtualization

ARTIFACTS

TestClient.java

TestServer.java

Recently, a customer reported an issue seeking assistance to virtualize their TCP backend service. Customer uses an Weblogic App Server which in turn talks to a TCP based backend service that was acting like a lookup engine to return some standard stuff like country codes, Zip code etc.

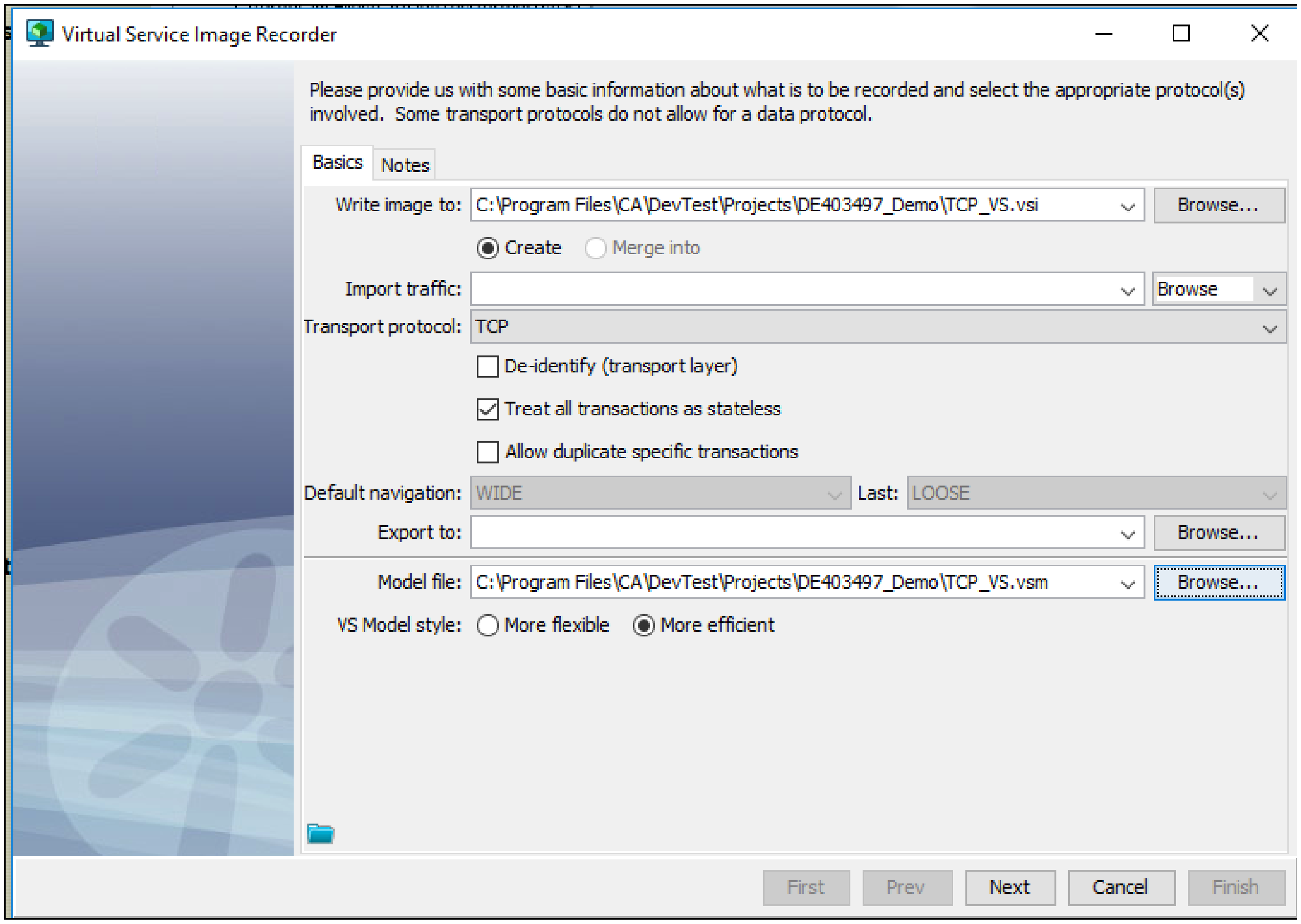
Their request and response contained a mix of plain text TCP headers and XML. An example is shown below:

REQUEST WITH HEADERS	RESPONSE WITH HEADERS
Content-length: 280 Content-type: application/xml nrfTransactionId: 15523212749834 <?xml version="1.0"?><nrf:NRFTransaction version="07.01.0001" RequestMethodName="raveGetCurrencyInfo" RequestMethodResponseName="raveGetCurrencyInfoResponse" transactionID="15523212749834" xmlns:nrf="http://ups.com/nrfServerInterface"><nrf:Request><nrf:raveGetCurrencyInfo originCode="US" destCode="US"/></nrf:Request></nrf:NRFTransaction>	HTTP/1.0 200 OK Content-Length:1200 Content-Type:application/xml <nrf:NRFTransaction xmlns:nrf="http://ups.com/nrfServerInterface" RequestMethodName="raveGetCurrencyInfo" RequestMethodResponseName="raveGetCurrencyInfoResponse" transactionID="15523212749834" version="07.01.0001"><nrf:Response><nrf:raveGetCurrencyInfoRe currencyCode="USD" currencyName="dollar" euroAllowed="0"/></nrf:raveGetCurrencyInfoResponse></nrf:Respo
Content-length: 280 Content-type: application/xml nrfTransactionId: 15523280765362 <?xml version="1.0"?><nrf:NRFTransaction version="07.01.0001" RequestMethodName="raveGetIsDutiable" RequestMethodResponseName="raveGetIsDutiableResponse" transactionID="15523280765362" xmlns:nrf="http://ups.com/nrfServerInterface"><nrf:Request><nrf:raveGetIsDutiable origCountry="CA" origPostalCode="" origCity="" destCountry="DE" destPostalCode="" destCity=""/></nrf:Request></nrf:NRFTransaction>	HTTP/1.0 200 OK Content-Length:1200 Content-Type:application/xml <nrf:NRFTransaction xmlns:nrf="http://ups.com/nrfServerInterface" RequestMethodName="raveGetIsDutiable" RequestMethodResponseName="raveGetIsDutiableResponse" transactionID="15523280765362" version="07.01.0001"><nrf:Response><nrf:raveGetIsDutiableRespo isDutiable="N"/></nrf:Response></nrf:NRFTransaction>

IN order to simulate customers BACKEND and CLIENT, couple of JAVA programs were written. Just start the TestServer,java and it will listen on port 1607. Once the server starts, the VSE recorder need to be started. Refer to screenshot below.

Once the recorder starts listening, invoke the JAVA Client code TestClient.java. You will be presented with options, just input the following options 1, 2, 3, 4 and 6. (DONT type option 5

RECORDING SCREENS:


Virtual Service Image Recorder

Please provide us with some basic information about what is to be recorded and select the appropriate protocol(s) involved. Some transport protocols do not allow for a data protocol.

Basics | Notes

Write image to: C:\Program Files\CA\DevTest\Projects\DE403497_Demo\TCP_VS.vsi

[✓ Create] Merge into

Import traffic:

Transport protocol: TCP

☐ De-identify (transport layer)

☑ Treat all transactions as stateless

☐ Allow duplicate specific transactions

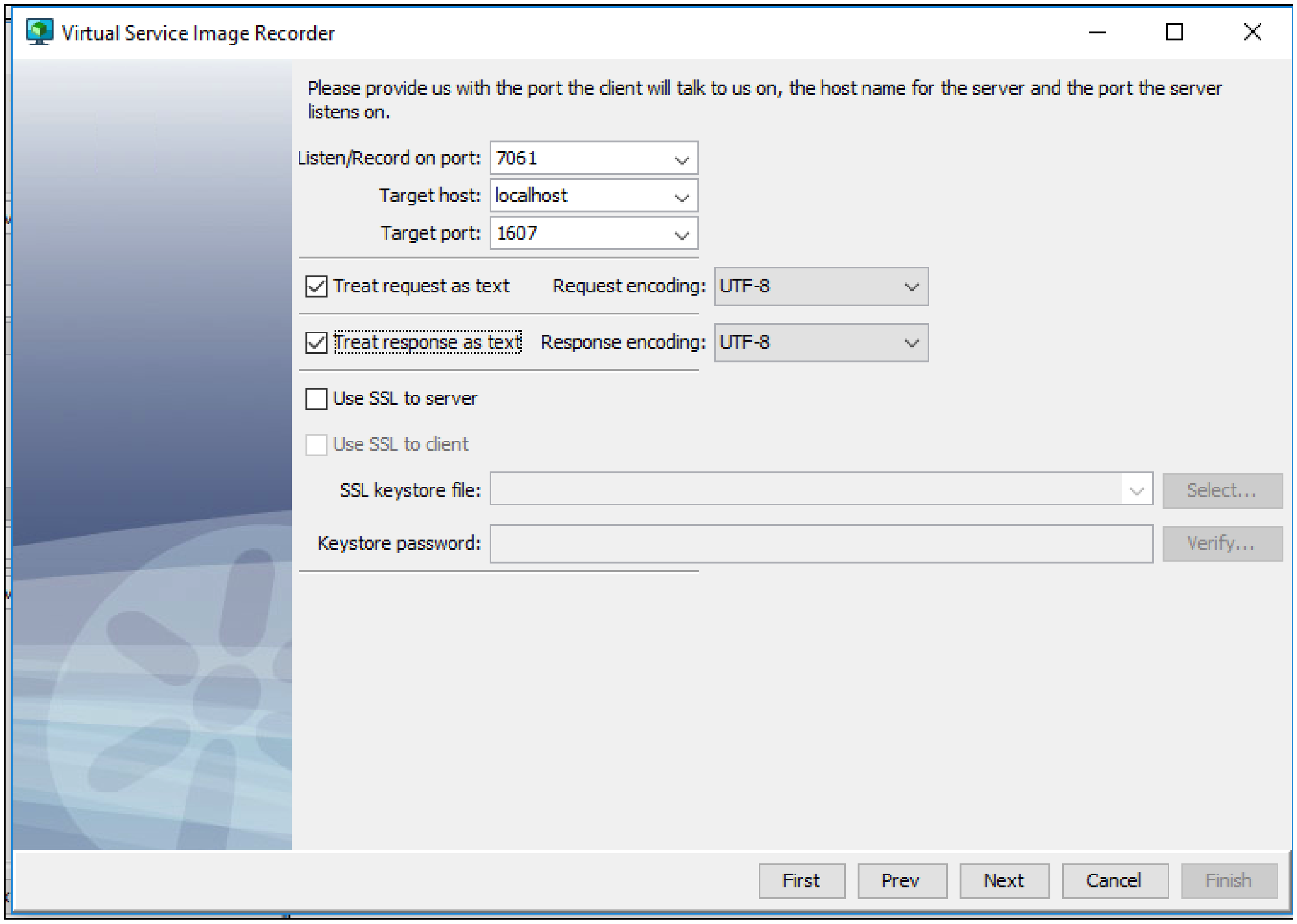
Default navigation: WIDE Last: LOOSE

Export to:

Model file: C:\Program Files\CA\DevTest\Projects\DE403497_Demo\TCP_VS.vsm

VS Model style: More flexible [✓ More efficient]

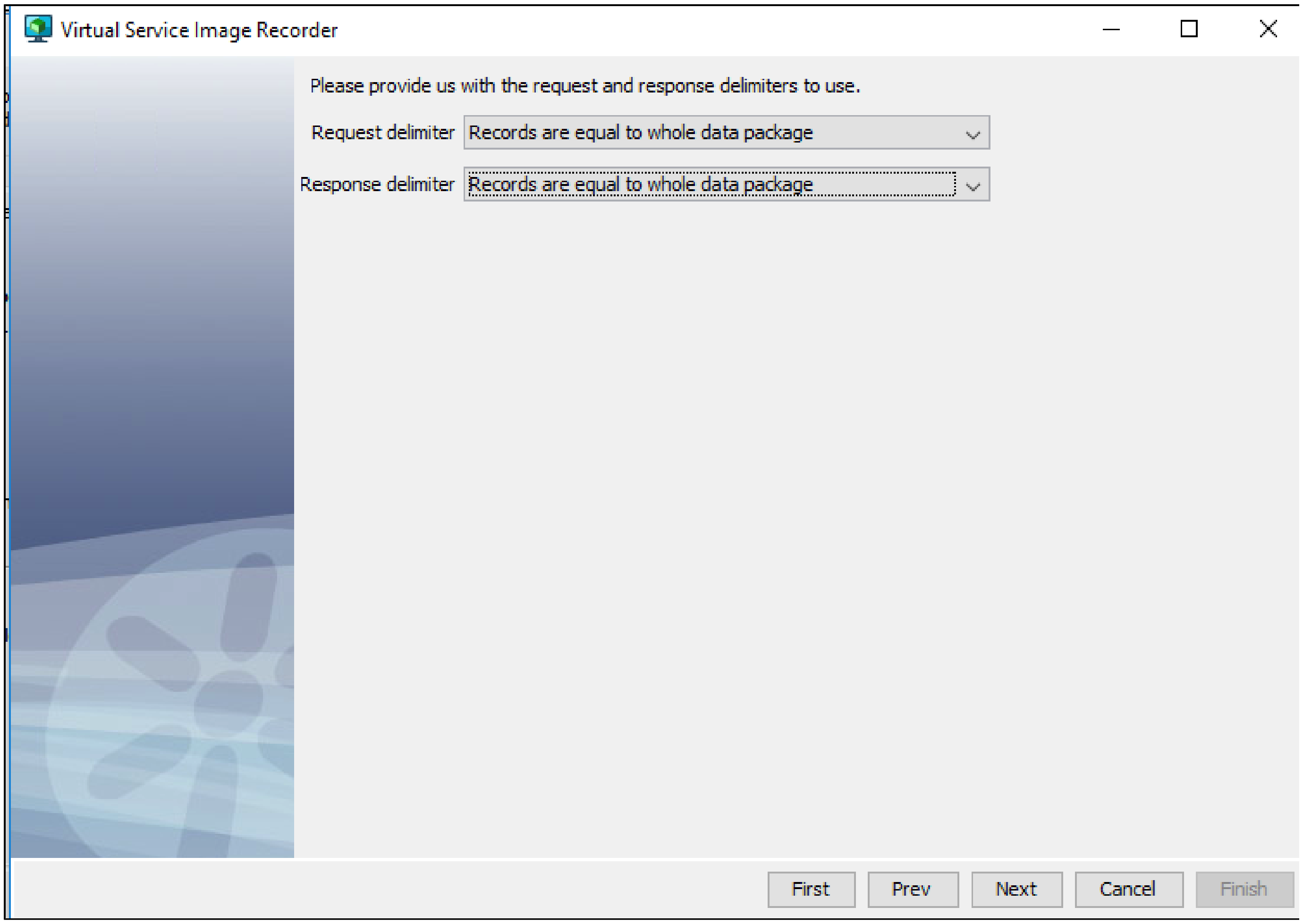
CHoose the ports. These are the ports used by JAVA Server and CLIENT program. If you want to change the ports, update the JAVA code



Please provide us with the port the client will talk to us on, the host name for the server and the port the server listens on.

Listen/Record on port: 7061
Target host: localhost
Target port: 1607
[✓] Treat request as text Request encoding: UTF-8
[✓] Treat response as text Response encoding: UTF-8
[ ] Use SSL to server
[ ] Use SSL to client
SSL keystore file:
Keystore password:

Treat the data package as a WHOLE. Thats how this customers BACKEND was behaving.

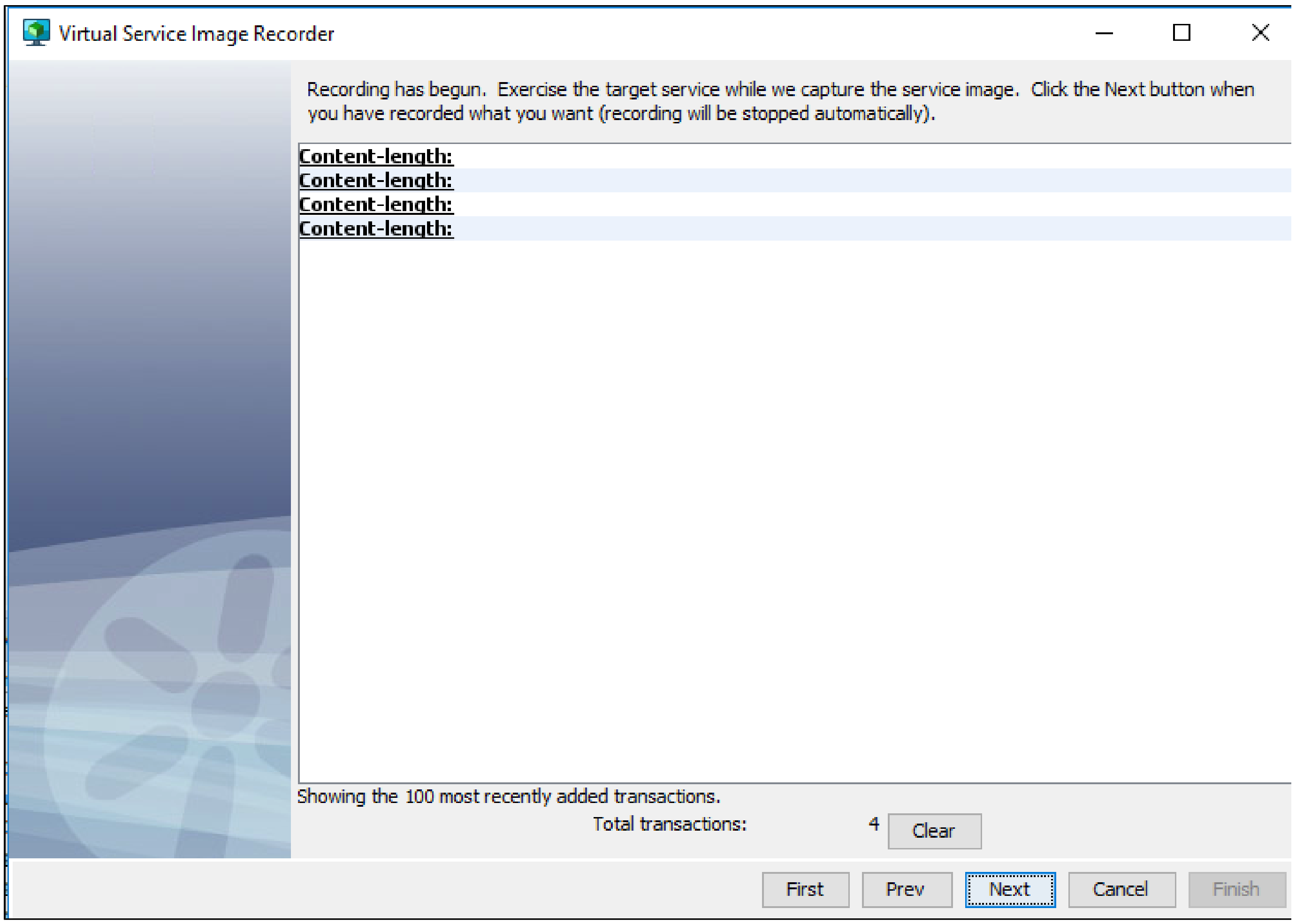


Please provide us with the request and response delimiters to use.

Request delimiter: Records are equal to whole data package

Response delimiter: Records are equal to whole data package

Now you got 4 req-res pairs recorded, Its time to check them once to see if we are good and move on to Data Protocol Handlers DPH



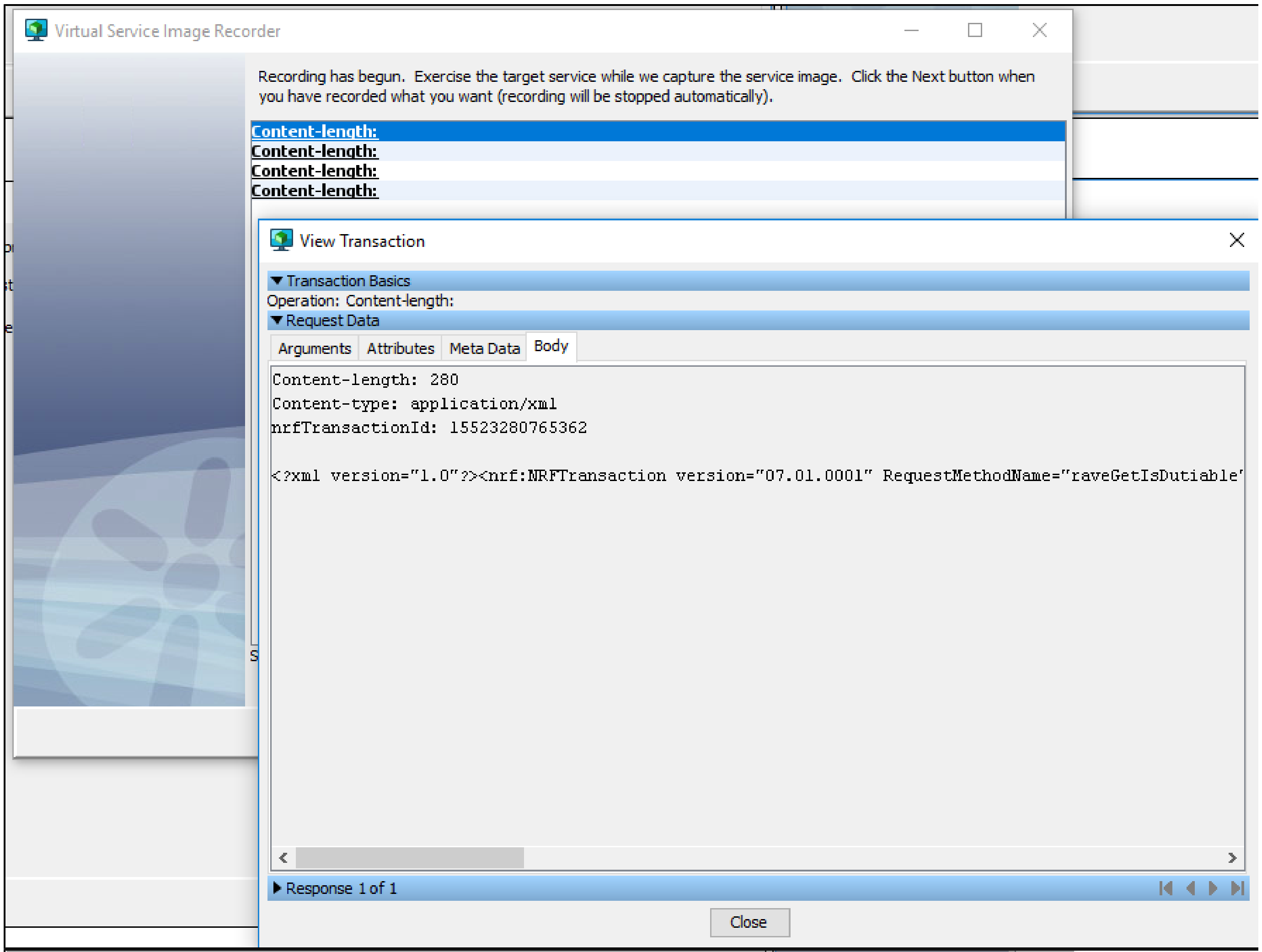
Recording has begun. Exercise the target service while we capture the service image. Click the Next button when you have recorded what you want (recording will be stopped automatically).

```
Content-length:
Content-length:
Content-length:
Content-length:
```

Showing the 100 most recently added transactions.

Total transactions: 4

Inspect the recorded REQUEST and RESPONSE pairs. Make sure the recording is good



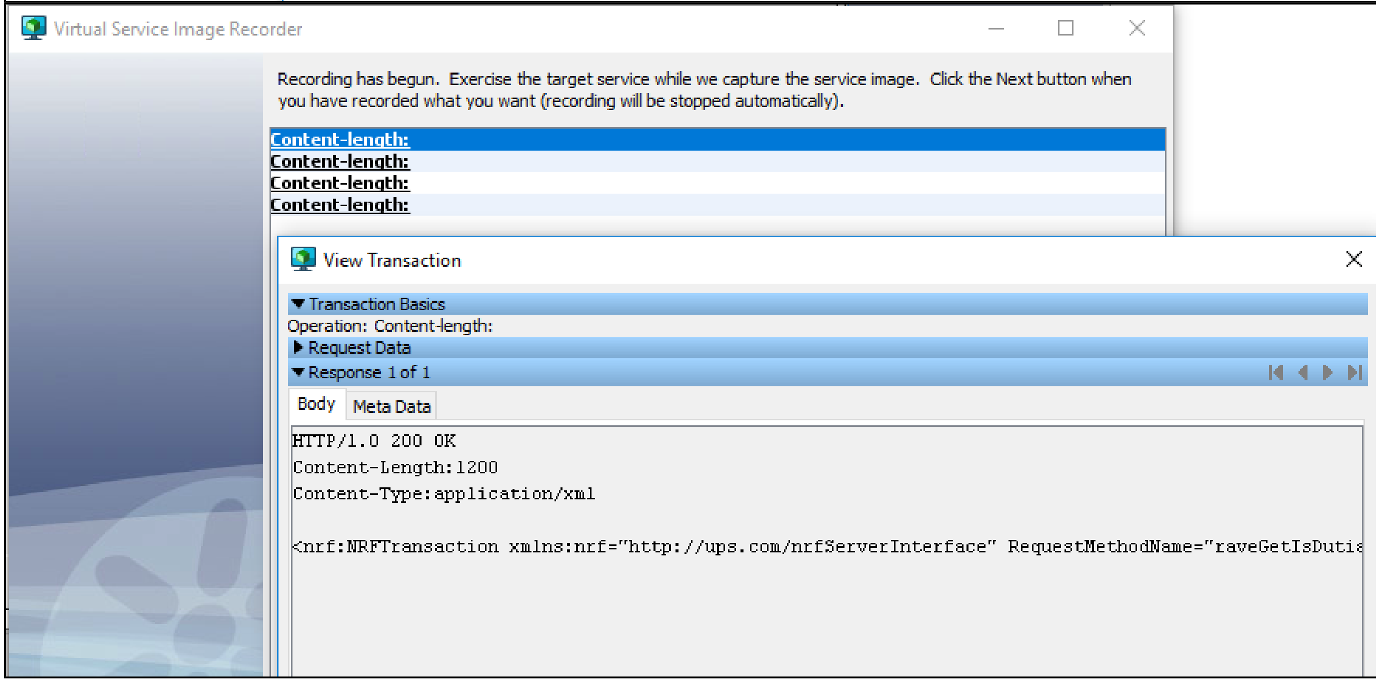
Recording has begun. Exercise the target service while we capture the service image. Click the Next button when you have recorded what you want (recording will be stopped automatically).

Content-length:
Content-length:
Content-length:
Content-length:

Operation: Content-length:

```
Content-length: 280
Content-type: application/xml
nrfTransactionId: 15523280765362

<?xml version="1.0"?><nrf:NRFTransaction version="07.01.0001" RequestMethodName="raveGetIsDutiable"
```



Recording has begun. Exercise the target service while we capture the service image. Click the Next button when you have recorded what you want (recording will be stopped automatically).

Content-length:
Content-length:
Content-length:
Content-length:

Operation: Content-length:

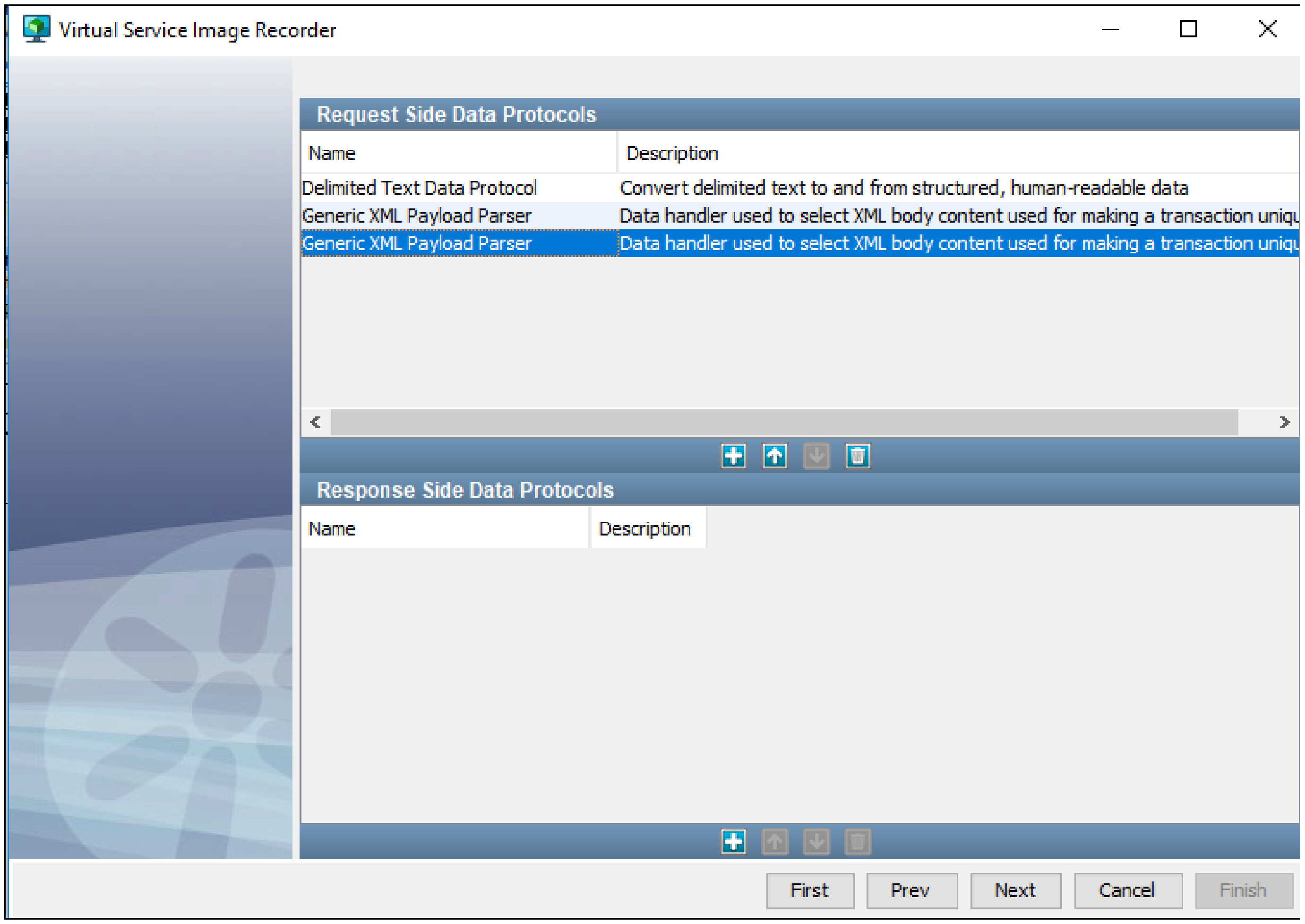
```
HTTP/1.0 200 OK
Content-Length:1200
Content-Type:application/xml

<nrf:NRFTransaction xmlns:nrf="http://ups.com/nrfServerInterface" RequestMethodName="raveGetIsDutia
```

IMPORTANT STEP: Request side data protocols. Since the REQUEST contains a mix of plain text (TCP Header) and XML, here is how we go about dissecting the request data.

1. **Demilited Text Data Protocol - To extract XML Payload from mixed data package**
2. **Generic XML Payload Parser - To assign extracted XML to Request Body**

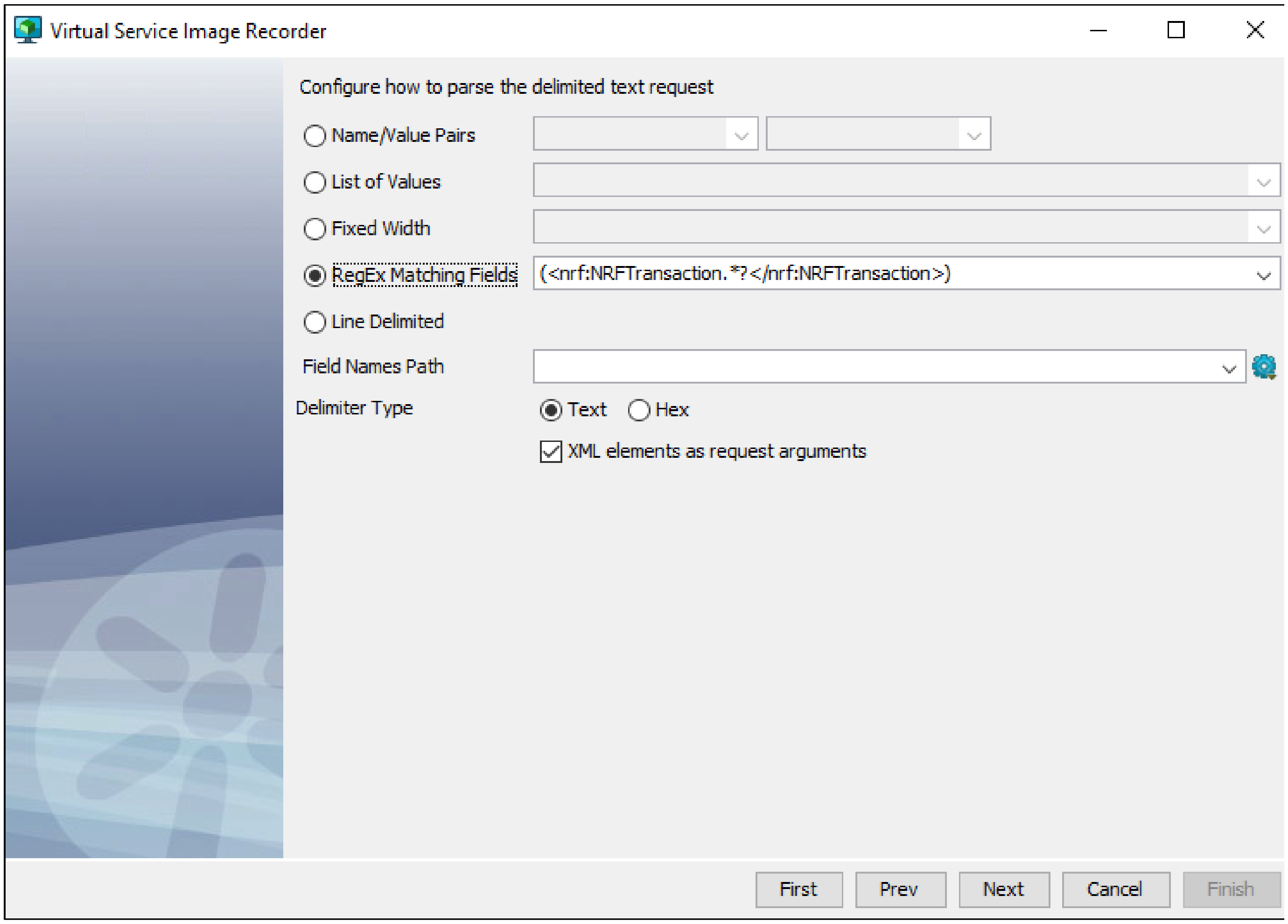
3. **Generic XML Payload Parser - To extract 'Operation' and 'Arguments' from Request Body**


Virtual Service Image Recorder
Request Side Data Protocols
Response Side Data Protocols
First
Prev
Next
Cancel
Finish

Name	Description
Delimited Text Data Protocol	Convert delimited text to and from structured, human-readable data
Generic XML Payload Parser	Data handler used to select XML body content used for making a transaction uniqu
Generic XML Payload Parser	Data handler used to select XML body content used for making a transaction uniqu

Name	Description

DPH 1 - Demilited Text Data Protocol - USED a Regex to just extract the XML part alone. Devtest assigns the value of the XML data to a variable called val1



Configure how to parse the delimited text request

- ○ Name/Value Pairs
- ○ List of Values
- ○ Fixed Width
- ● RegEx Matching Fields (<nrf:NRFTransaction.*?</nrf:NRFTransaction>)
- ○ Line Delimited

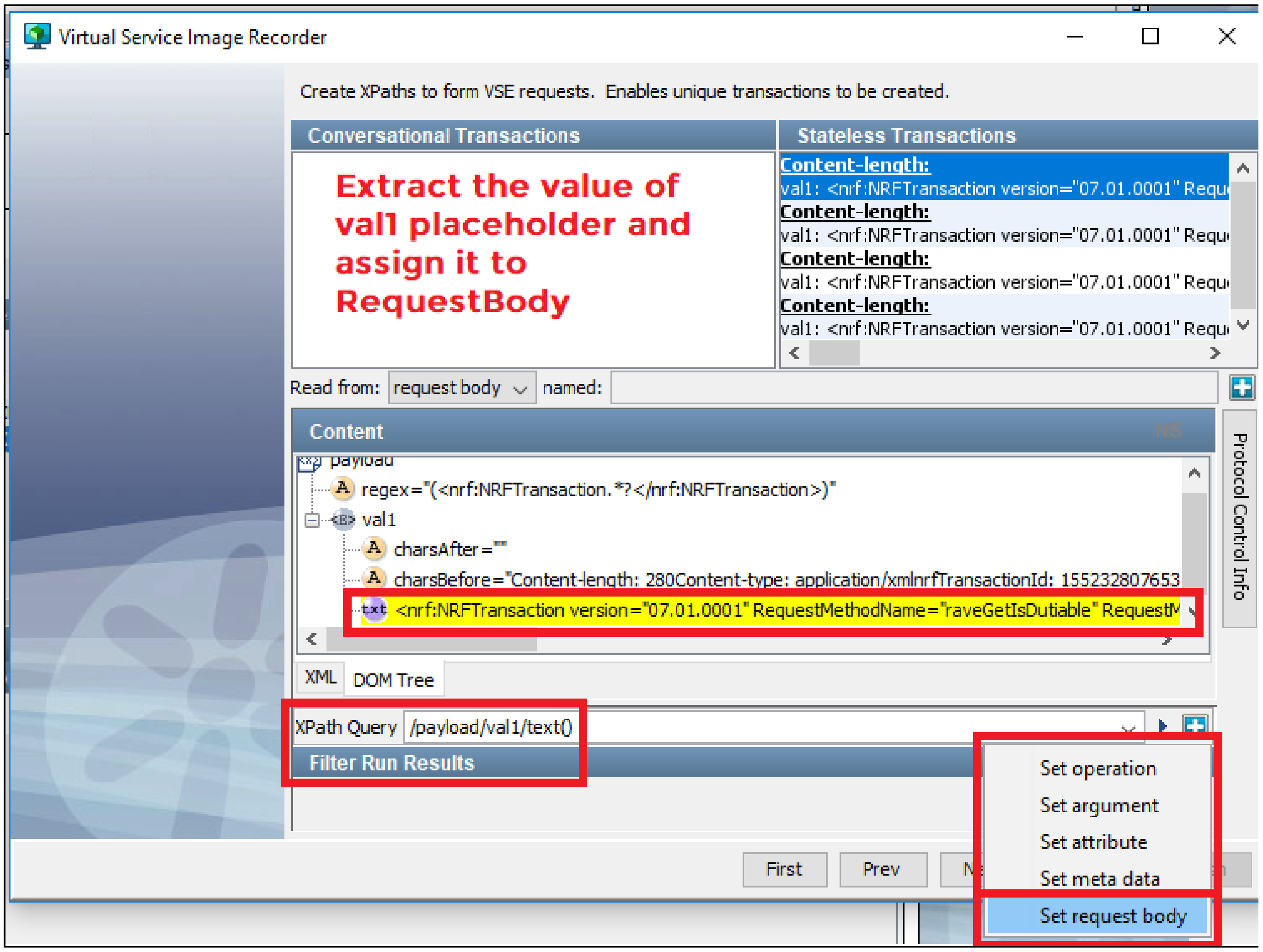
Field Names Path:

Delimiter Type: [✓ Text] Hex

☑ XML elements as request arguments

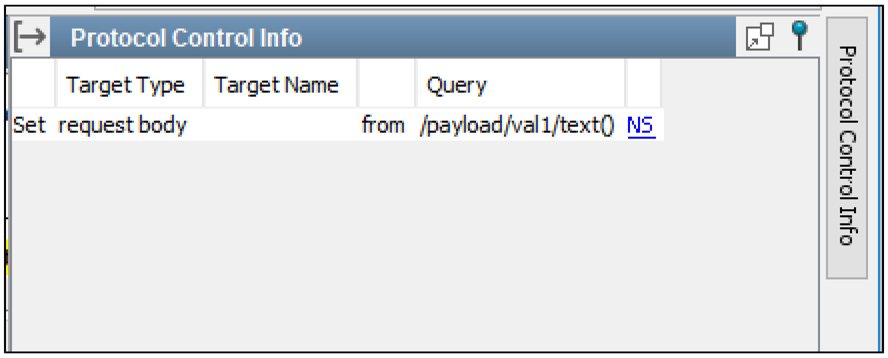
DPH 2 - Generic XML Payload Parser - All, we now have from previous step is val1. This value is better assigned to 'Request Body' so that we can extract 'Operation' and 'Arguments' in next step

When you click the + sign and add set the 'RequestBody', you wont see the assignment anywhere. You need to click the vertical button with label 'Protocol Control Info' to see all the assigned variables


Virtual Service Image Recorder
Create XPaths to form VSE requests. Enables unique transactions to be created.
Conversational Transactions
Stateless Transactions
Extract the value of val1 placeholder and assign it to RequestBody
Read from: request body
named:
Content
Protocol Control Info
XML
DOM Tree
XPath Query /payload/val1/text()
Filter Run Results
Set operation
Set argument
Set attribute
Set meta data
Set request body
First
Prev

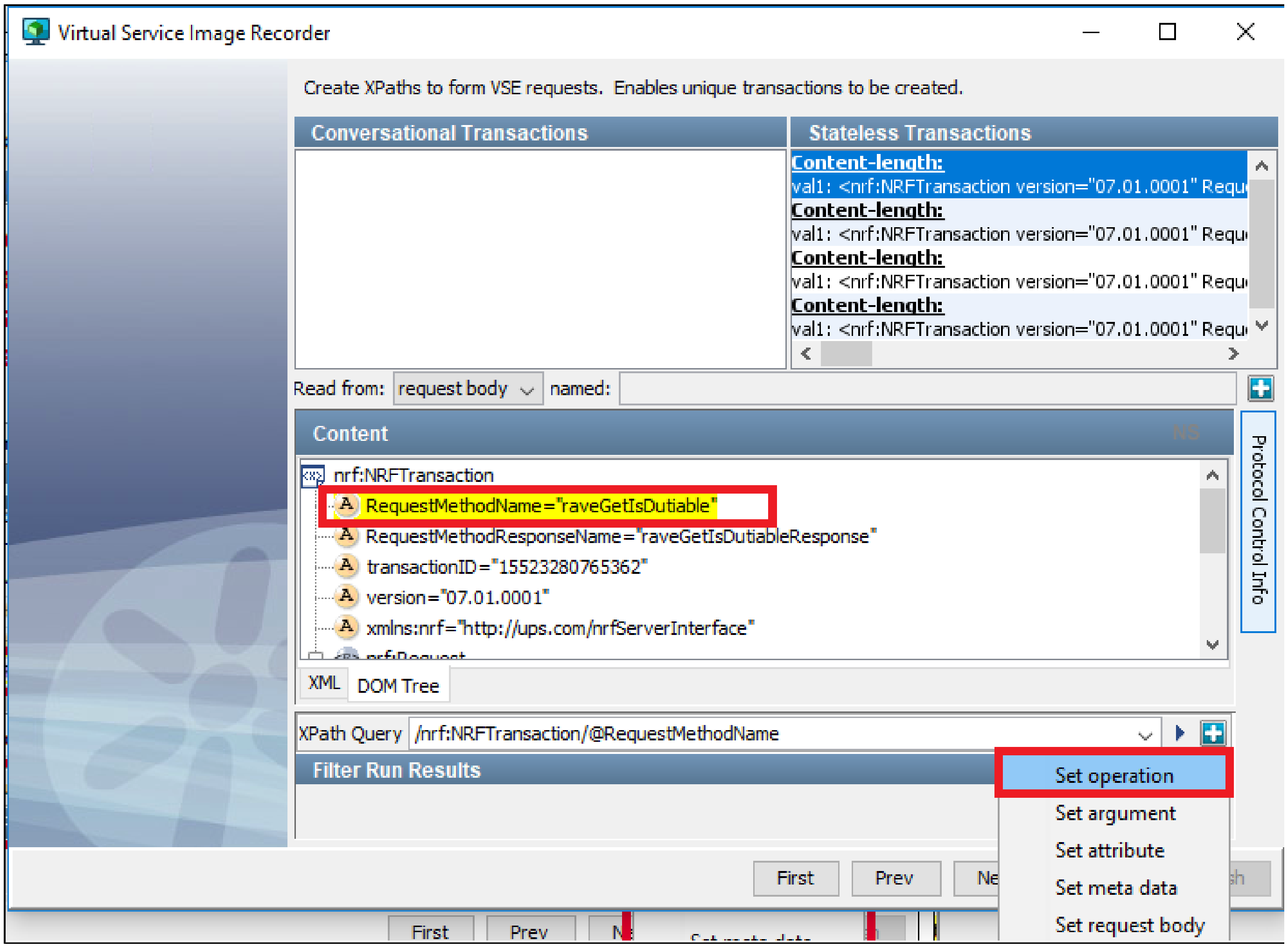
```
Content-length:
val1: <nrf:NRFTransaction version="07.01.0001" Requ
Content-length:
val1: <nrf:NRFTransaction version="07.01.0001" Requ
Content-length:
val1: <nrf:NRFTransaction version="07.01.0001" Requ
Content-length:
val1: <nrf:NRFTransaction version="07.01.0001" Requ
```

```
payload
  regex="(<nrf:NRFTransaction.*?</nrf:NRFTransaction>)"
  val1
    charsAfter=""
    charsBefore="Content-length: 280Content-type: application/xmlnrfTransactionId: 155232807653
    <nrf:NRFTransaction version="07.01.0001" RequestMethodName="raveGetIsDutiable" RequestM
```



	Target Type	Target Name		Query	
Set	request body		from	/payload/val1/text()	NS

DPH 3 - Generic XML Payload Parser - In this step, we identify and extract 'Operation' and 'Arguments'. If the different captured req-resp pairs have different arguments, make sure you identify all the possible arguments.


Virtual Service Image Recorder

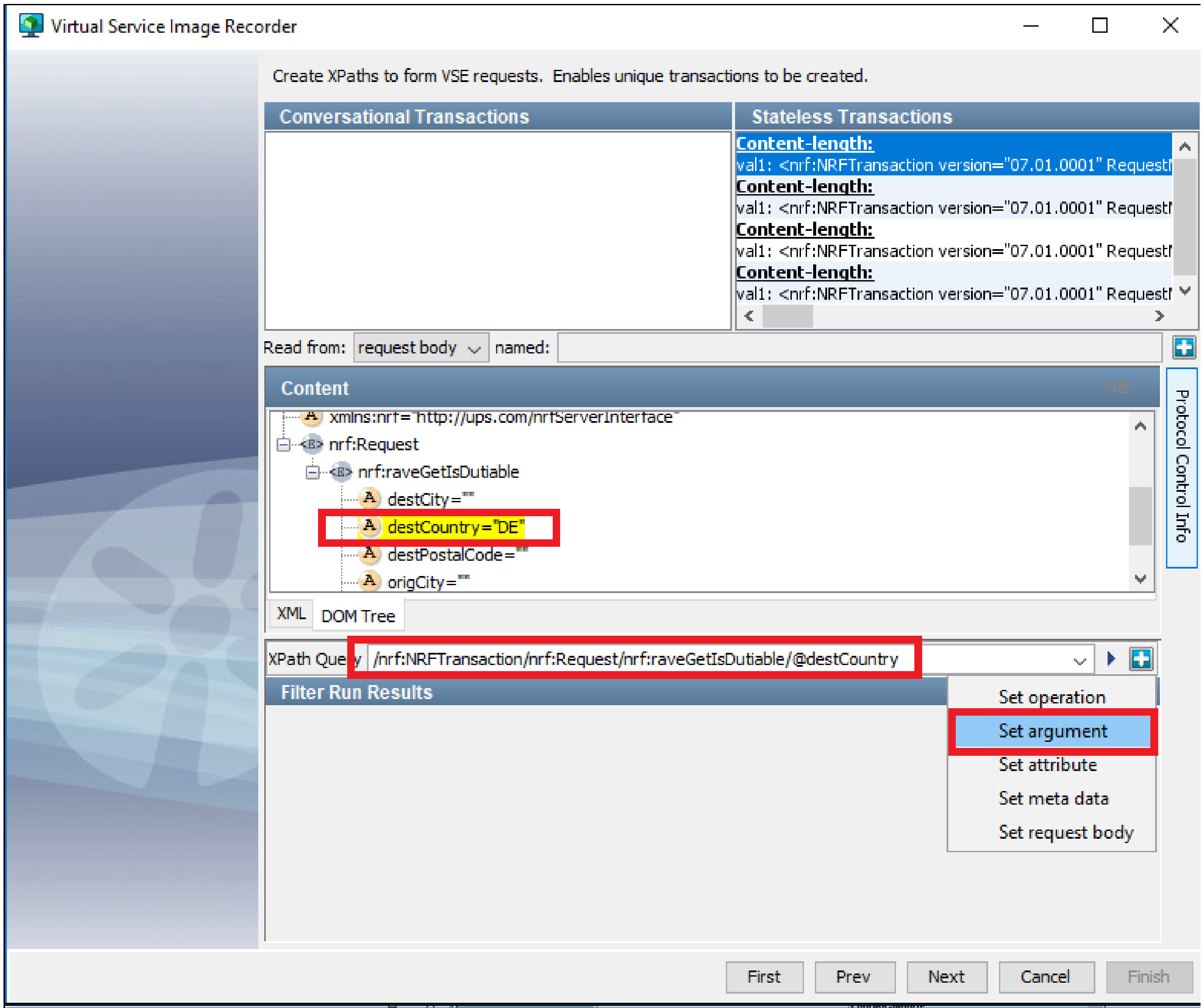
Create XPaths to form VSE requests. Enables unique transactions to be created.

Content-length:
val1: <nrf:NRFTransaction version="07.01.0001" Requ
Content-length:
val1: <nrf:NRFTransaction version="07.01.0001" Requ
Content-length:
val1: <nrf:NRFTransaction version="07.01.0001" Requ
Content-length:
val1: <nrf:NRFTransaction version="07.01.0001" Requ

Read from: request body named:

nrf:NRFTransaction
RequestMethodName="raveGetIsDutiable"
RequestMethodResponseName="raveGetIsDutiableResponse"
transactionID="15523280765362"
version="07.01.0001"
xmlns:nrf="http://ups.com/nrfServerInterface"

XPath Query /nrf:NRFTransaction/@RequestMethodName

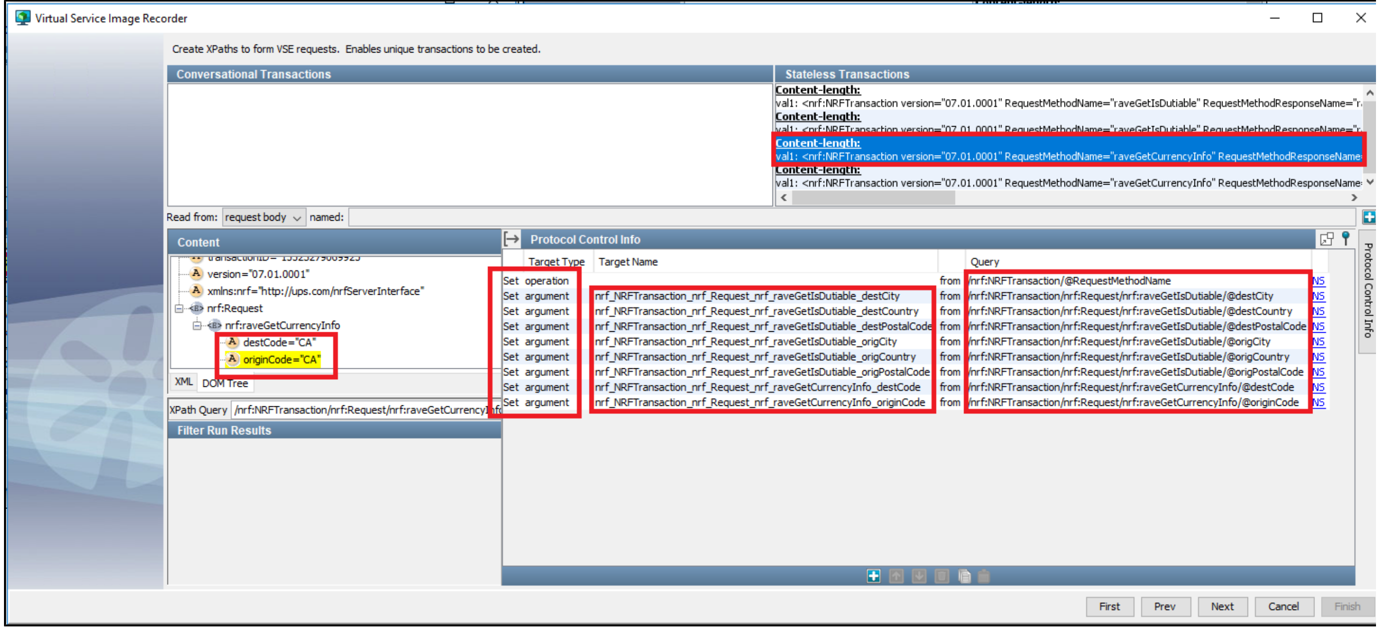


Create XPaths to form VSE requests. Enables unique transactions to be created.

```
Content-length:
val1: <nrf:NRFTransaction version="07.01.0001" Request
Content-length:
val1: <nrf:NRFTransaction version="07.01.0001" Request
Content-length:
val1: <nrf:NRFTransaction version="07.01.0001" Request
Content-length:
val1: <nrf:NRFTransaction version="07.01.0001" Request
```

```
xmlns:nrf="http://ups.com/nrfServerInterface"
nrf:Request
  nrf:raveGetIsDutiable
    destCity=""
    destCountry="DE"
    destPostalCode=""
    origCity=""
```

XPath Query: /nrf:NRFTransaction/nrf:Request/nrf:raveGetIsDutiable/@destCountry



Create XPaths to form VSE requests. Enables unique transactions to be created.

```
Content-length:
val1: <nrf:NRFTransaction version="07.01.0001" RequestMethodName="raveGetIsDutiable" RequestMethodResponseName="r
Content-length:
val1: <nrf:NRFTransaction version="07.01.0001" RequestMethodName="raveGetIsDutiable" RequestMethodResponseName="r
Content-length:
val1: <nrf:NRFTransaction version="07.01.0001" RequestMethodName="raveGetCurrencyInfo" RequestMethodResponseName
Content-length:
val1: <nrf:NRFTransaction version="07.01.0001" RequestMethodName="raveGetCurrencyInfo" RequestMethodResponseName
```

```
version="07.01.0001"
xmlns:nrf="http://ups.com/nrfServerInterface"
nrf:Request
  nrf:raveGetCurrencyInfo
    destCode="CA"
    originCode="CA"
```

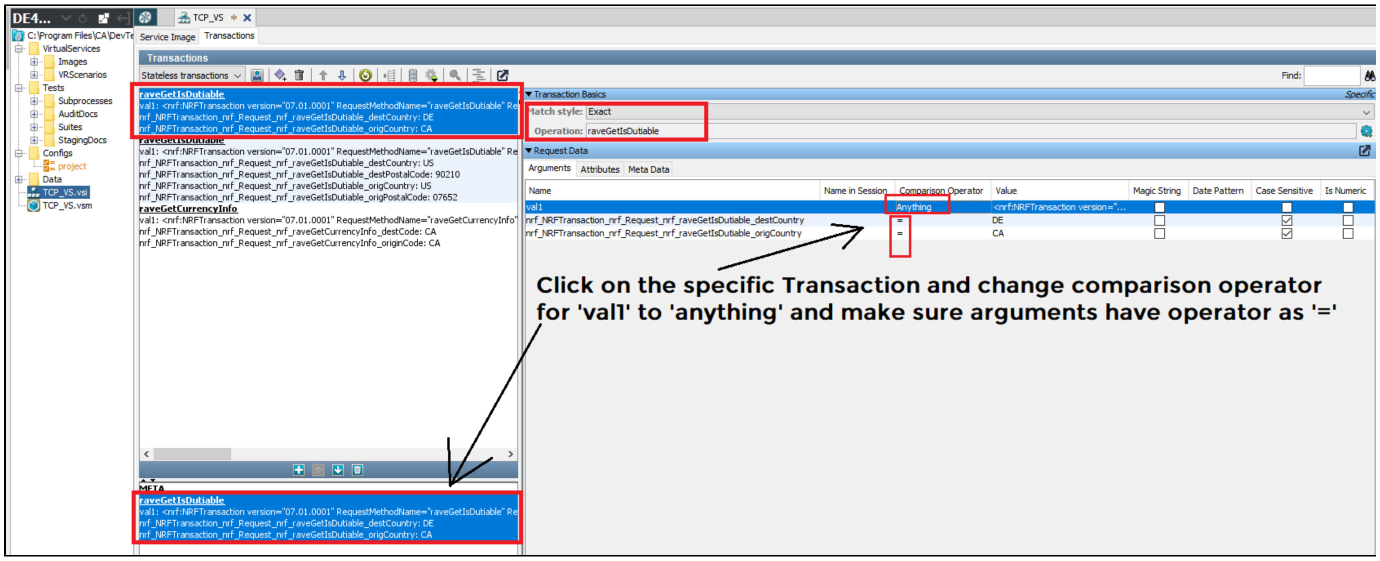
XPath Query: /nrf:NRFTransaction/nrf:Request/nrf:raveGetCurrencyInfo

	Target Type	Target Name		Query	
Set	operation		from	/nrf:NRFTransaction/@RequestMethodName	NS
Set	argument	nrf_NRFTransaction_nrf_Request_nrf_raveGetIsDutiable_destCity	from	/nrf:NRFTransaction/nrf:Request/nrf:raveGetIsDutiable/@destCity	NS
Set	argument	nrf_NRFTransaction_nrf_Request_nrf_raveGetIsDutiable_destCountry	from	/nrf:NRFTransaction/nrf:Request/nrf:raveGetIsDutiable/@destCountry	NS
Set	argument	nrf_NRFTransaction_nrf_Request_nrf_raveGetIsDutiable_destPostalCode	from	/nrf:NRFTransaction/nrf:Request/nrf:raveGetIsDutiable/@destPostalCode	NS
Set	argument	nrf_NRFTransaction_nrf_Request_nrf_raveGetIsDutiable_origCity	from	/nrf:NRFTransaction/nrf:Request/nrf:raveGetIsDutiable/@origCity	NS
Set	argument	nrf_NRFTransaction_nrf_Request_nrf_raveGetIsDutiable_origCountry	from	/nrf:NRFTransaction/nrf:Request/nrf:raveGetIsDutiable/@origCountry	NS
Set	argument	nrf_NRFTransaction_nrf_Request_nrf_raveGetIsDutiable_origPostalCode	from	/nrf:NRFTransaction/nrf:Request/nrf:raveGetIsDutiable/@origPostalCode	NS
Set	argument	nrf_NRFTransaction_nrf_Request_nrf_raveGetCurrencyInfo_destCode	from	/nrf:NRFTransaction/nrf:Request/nrf:raveGetCurrencyInfo/@destCode	NS
Set	argument	nrf_NRFTransaction_nrf_Request_nrf_raveGetCurrencyInfo_originCode	from	/nrf:NRFTransaction/nrf:Request/nrf:raveGetCurrencyInfo/@originCode	NS

With this screen, recording eizard enfs and you would click FINISH.

Open the VSI file and change the comparison operators as shown below. We shouldnt be bothered about what comes in val1, hence we choose operation 'Anything'. Do this for all SPECIFIC transactions. META transactions anyway wouldnt have EXACT tolerance.

NOTE: Depending on your need, you can choose among Exact, Signature, Operation for SPECIFIC transactions.



raveGetIsDutiable
val1: <nrf:NRFTransaction version="07.01.0001" RequestMethodName="raveGetIsDutiable" Re
nrf_NRFTransaction_nrf_Request_nrf_raveGetIsDutiable_destCountry: DE
nrf_NRFTransaction_nrf_Request_nrf_raveGetIsDutiable_origCountry: CA

raveGetIsDutiable
val1: <nrf:NRFTransaction version="07.01.0001" RequestMethodName="raveGetIsDutiable" Re
nrf_NRFTransaction_nrf_Request_nrf_raveGetIsDutiable_destCountry: US
nrf_NRFTransaction_nrf_Request_nrf_raveGetIsDutiable_destPostalCode: 90210
nrf_NRFTransaction_nrf_Request_nrf_raveGetIsDutiable_origCountry: US
nrf_NRFTransaction_nrf_Request_nrf_raveGetIsDutiable_origPostalCode: 07652

raveGetCurrencyInfo
val1: <nrf:NRFTransaction version="07.01.0001" RequestMethodName="raveGetCurrencyInfo"
nrf_NRFTransaction_nrf_Request_nrf_raveGetCurrencyInfo_destCode: CA
nrf_NRFTransaction_nrf_Request_nrf_raveGetCurrencyInfo_originCode: CA

Match style: Exact
Operation: raveGetIsDutiable

Name	Name in Session	Comparison Operator	Value	Magic String	Date Pattern	Case Sensitive	Is Numeric
val1		Anything	<nrf:NRFTransaction version="…	☐		☐	☐
nrf_NRFTransaction_nrf_Request_nrf_raveGetIsDutiable_destCountry		=	DE	☐		✓	☐
nrf_NRFTransaction_nrf_Request_nrf_raveGetIsDutiable_origCountry		=	CA	☐		✓	☐

Click on the specific Transaction and change comparison operator for 'val1' to 'anything' and make sure arguments have operator as '='

META

raveGetIsDutiable
val1: <nrf:NRFTransaction version="07.01.0001" RequestMethodName="raveGetIsDutiable" Re
nrf_NRFTransaction_nrf_Request_nrf_raveGetIsDutiable_destCountry: DE
nrf_NRFTransaction_nrf_Request_nrf_raveGetIsDutiable_origCountry: CA

You are all set, just deploy the VSM.

<u>TESTING YOUR VIRTUAL SERVICE</u>

1. Just import the MAR file to an existing project or create a new project from the MAR file.

t821.mar

2. Deploy the Virtual Service t821

3. You should notice that it starts listening on port 7061

4. MAke sure you stop the JAVA TestServer program

5. INvoke JAVA TestClient program and no need to make any PORT changes as your VS will stand in for JAVA TestServer

6. Invoke the same options 1, 2, 3, 4 and check the response