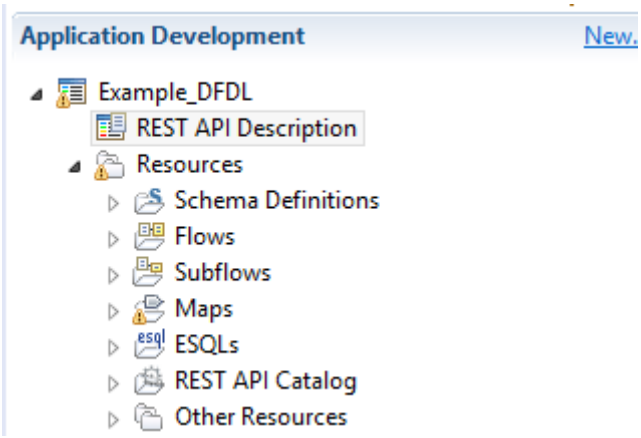
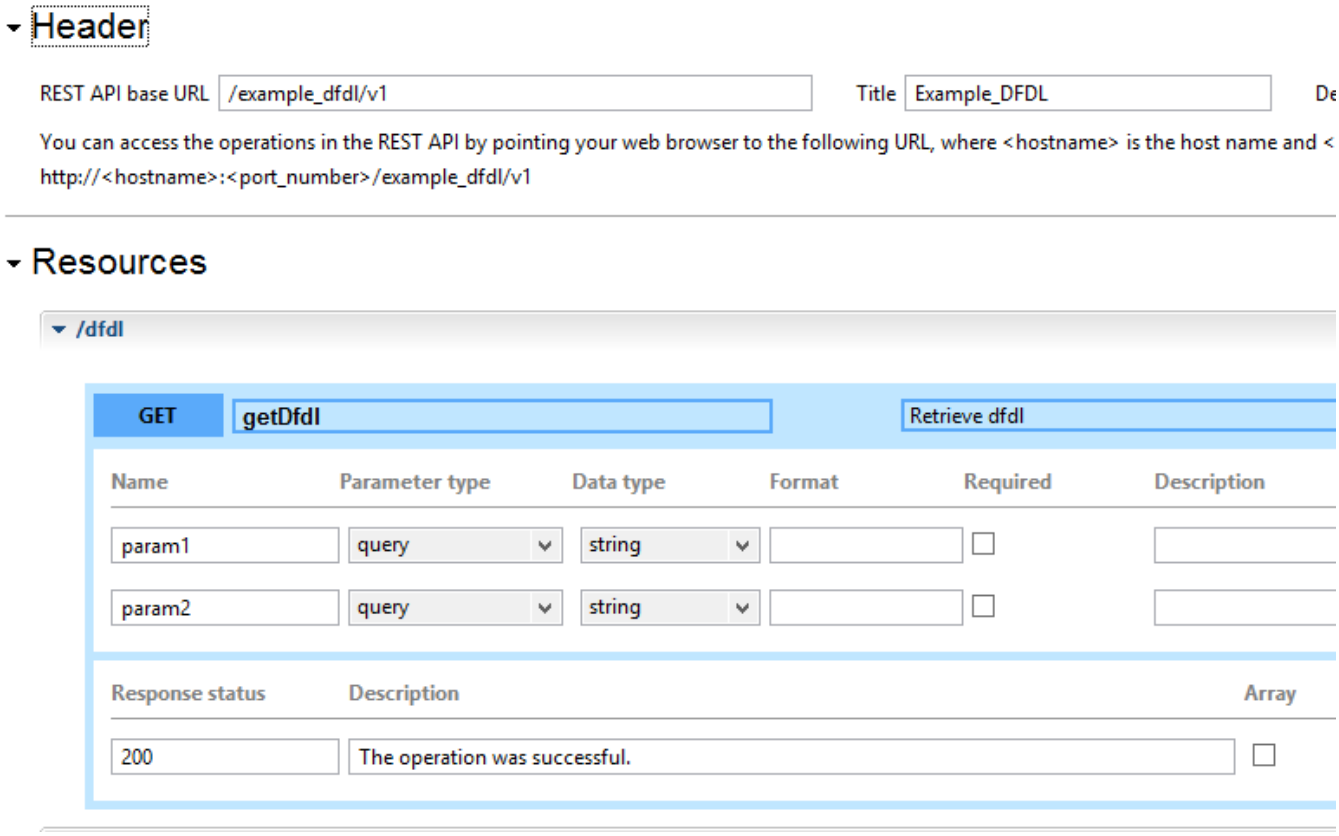
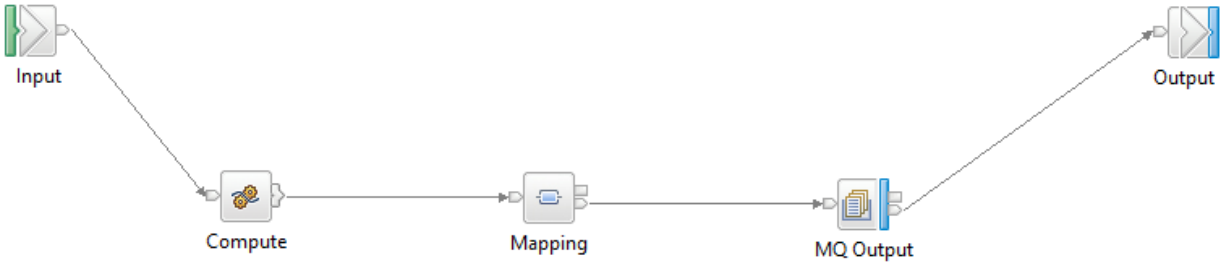
Código fuente: [Example_DFDL.zip]

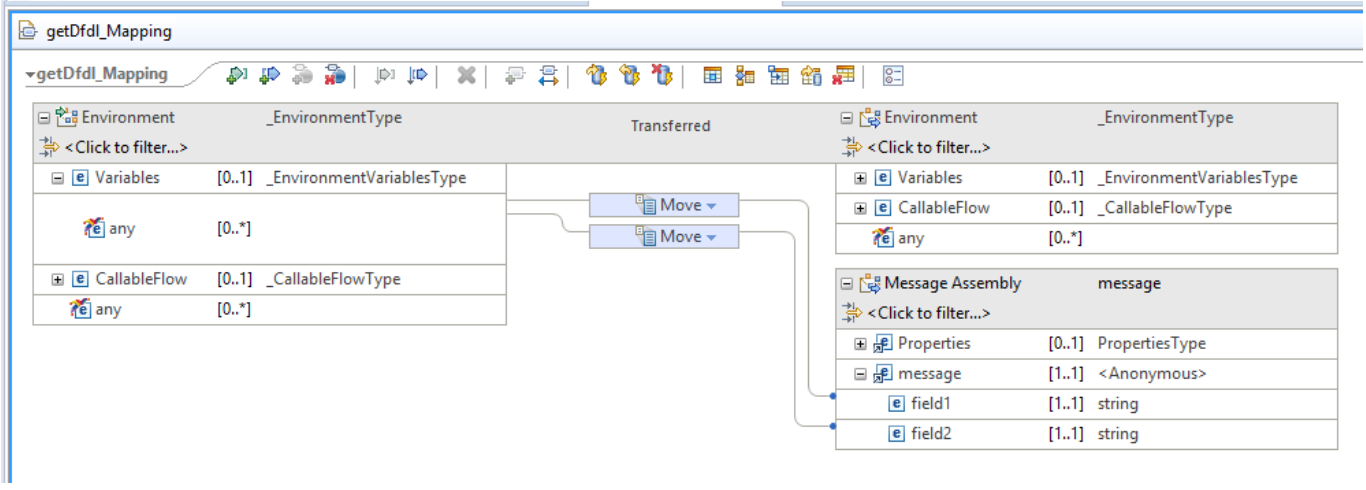


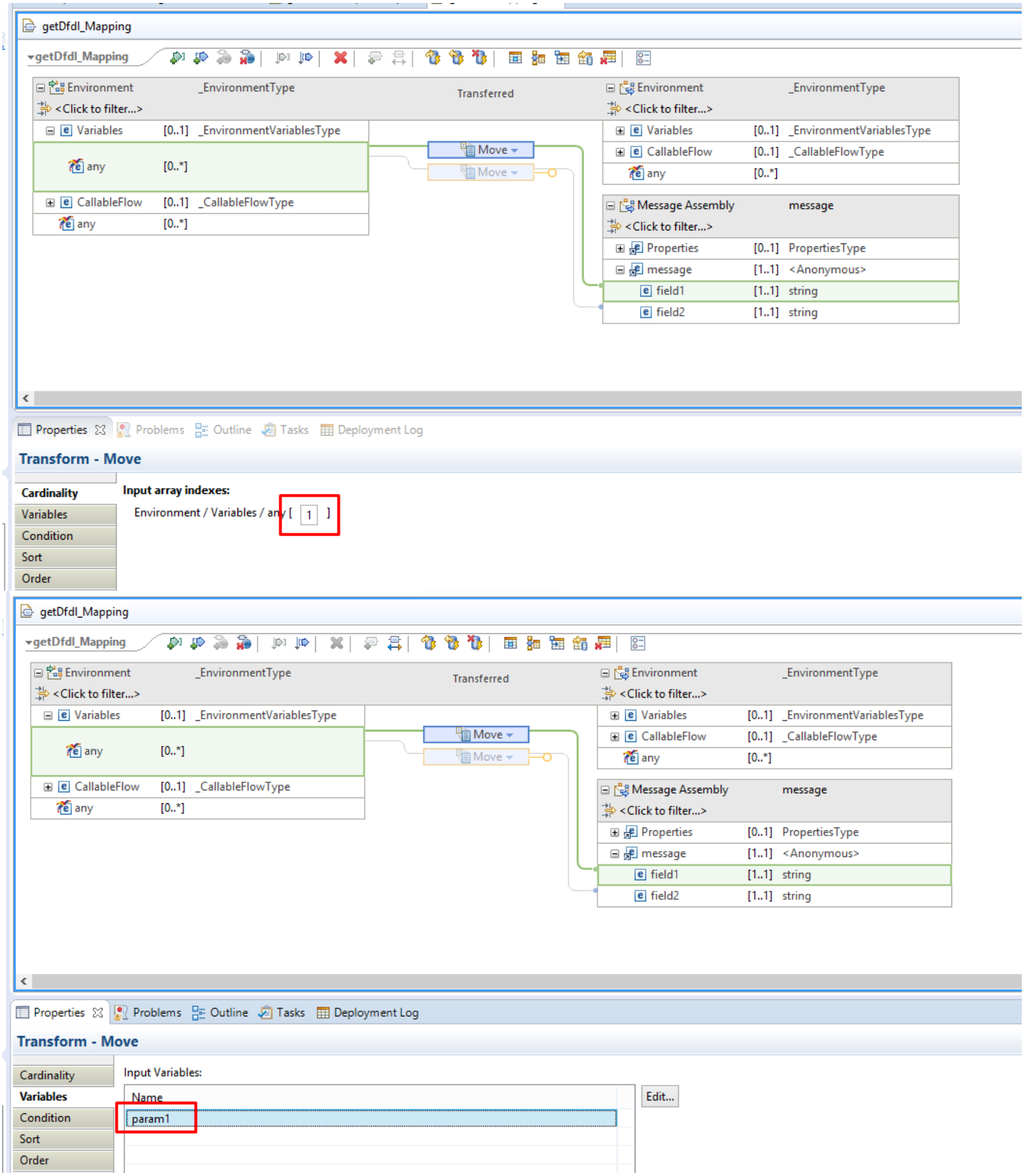





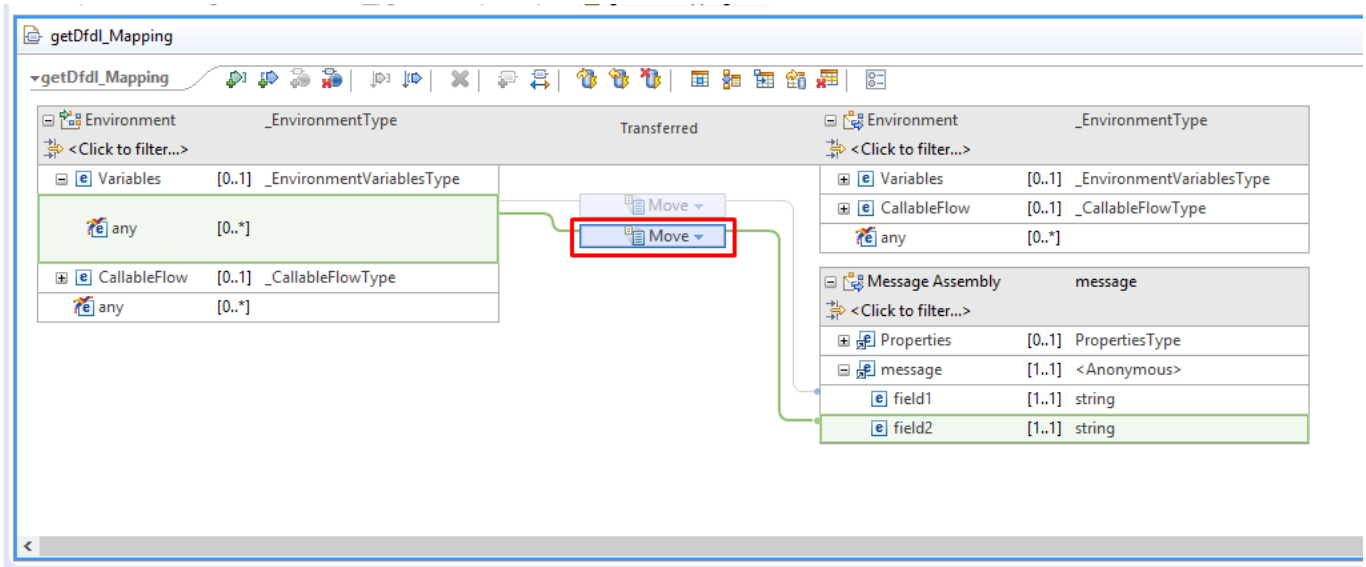

Example_DFDL | getDfdl.subflow | getDfdl_Compute.esql

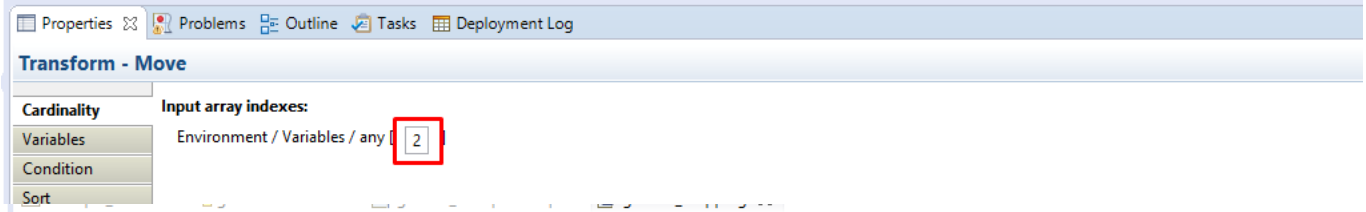
```
1 CREATE COMPUTE MODULE getDfdl_Compute
2     CREATE FUNCTION Main() RETURNS BOOLEAN
3     BEGIN
4
5         SET Environment.Variables.param1 = InputLocalEnvironment.REST.Input.Parameters.param1;
6         SET Environment.Variables.param2 = InputLocalEnvironment.REST.Input.Parameters.param2;
7
8         --SET OutputRoot.JSON.Data.mensaje.campo1 = Environment.Variables.param1;
9         --SET OutputRoot.JSON.Data.mensaje.campo2 = Environment.Variables.param2;
10
11        RETURN TRUE;
12    END;
13
14 END MODULE;
15
```

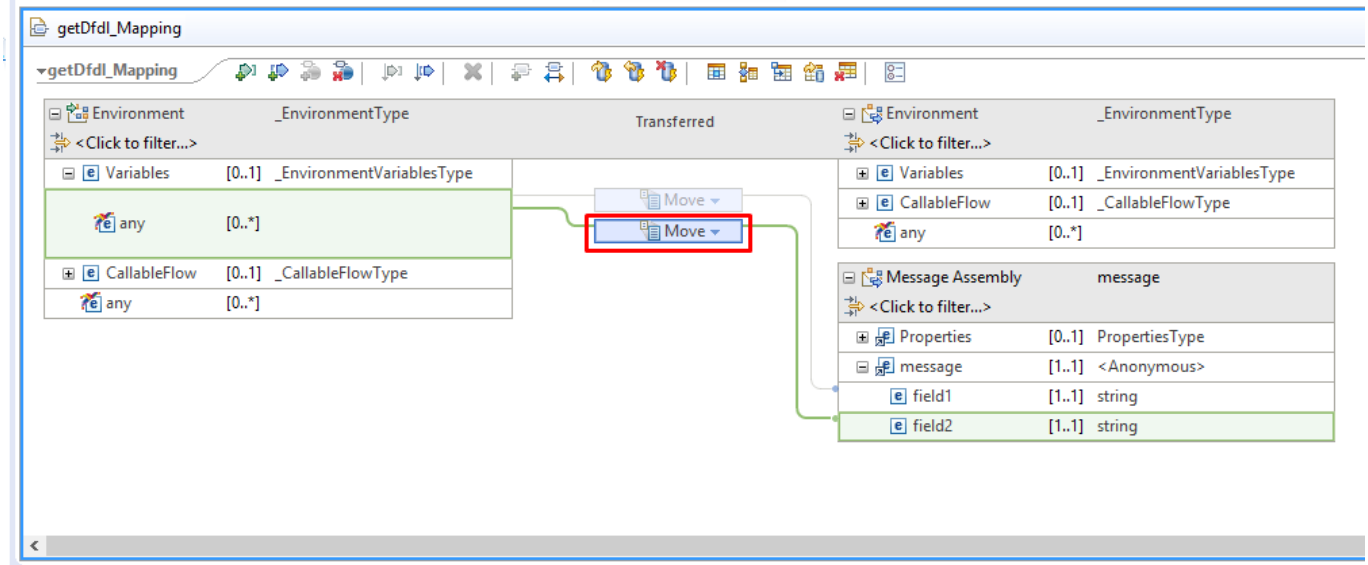


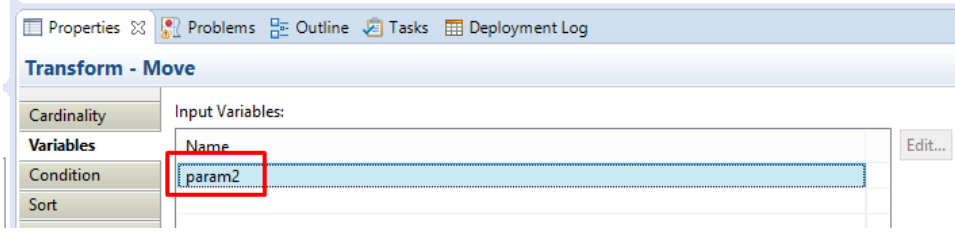

getDfdl_Mapping
Environment _EnvironmentType
<Click to filter...>
Variables [0..1] _EnvironmentVariablesType
any [0..*]
CallableFlow [0..1] _CallableFlowType
any [0..*]
Transferred
Move
Move
Environment _EnvironmentType
<Click to filter...>
Variables [0..1] _EnvironmentVariablesType
CallableFlow [0..1] _CallableFlowType
any [0..*]
Message Assembly message
<Click to filter...>
Properties [0..1] PropertiesType
message [1..1] <Anonymous>
field1 [1..1] string
field2 [1..1] string
Properties
Problems
Outline
Tasks
Deployment Log
Transform - Move
Cardinality
Variables
Condition
Sort
Order
Input array indexes:
Environment / Variables / any [ 1 ]

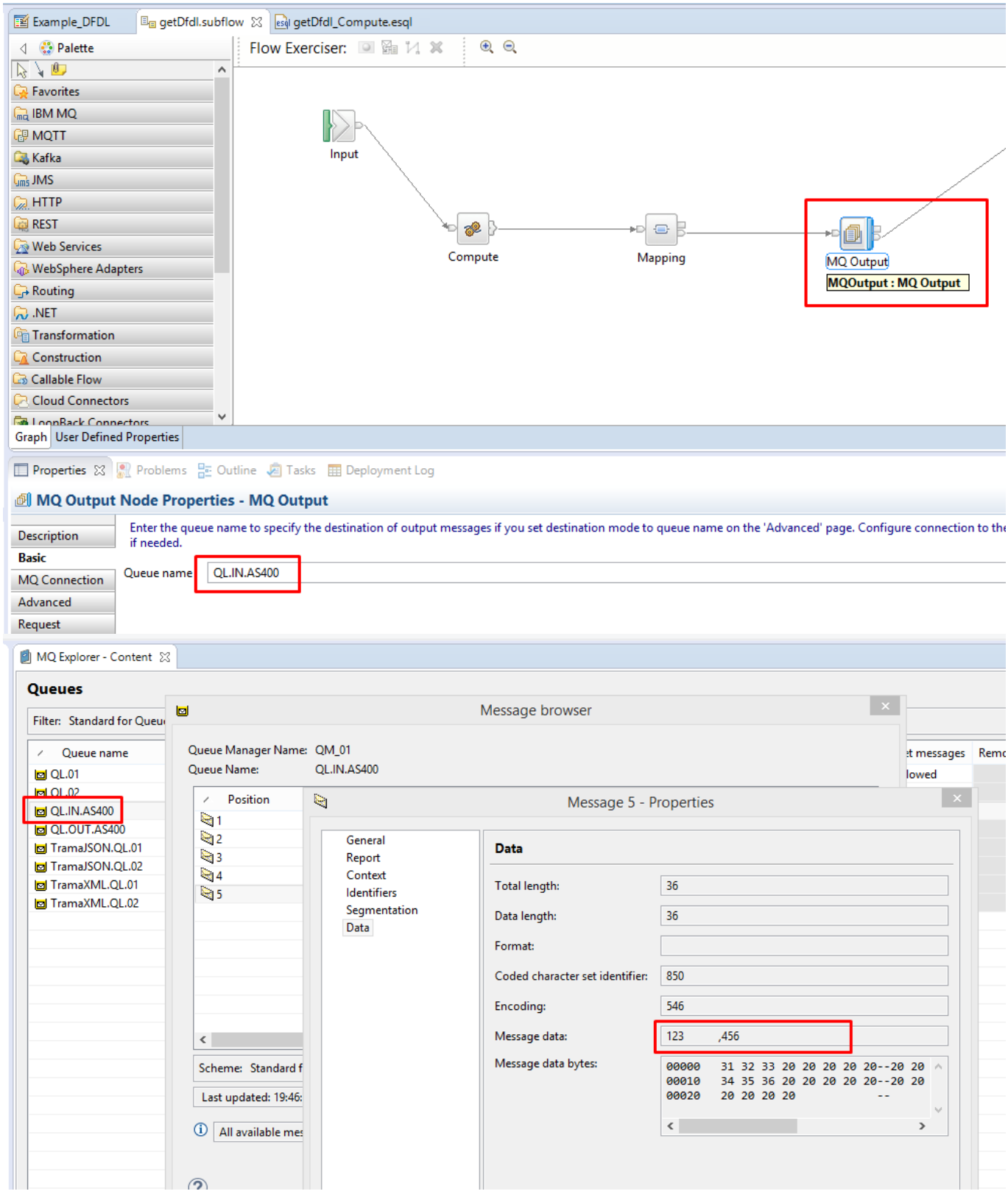

getDfdl_Mapping
Environment _EnvironmentType
<Click to filter...>
Variables [0..1] _EnvironmentVariablesType
any [0..*]
CallableFlow [0..1] _CallableFlowType
any [0..*]
Transferred
Move
Move
Environment _EnvironmentType
<Click to filter...>
Variables [0..1] _EnvironmentVariablesType
CallableFlow [0..1] _CallableFlowType
any [0..*]
Message Assembly message
<Click to filter...>
Properties [0..1] PropertiesType
message [1..1] <Anonymous>
field1 [1..1] string
field2 [1..1] string
Properties
Problems
Outline
Tasks
Deployment Log
Transform - Move
Cardinality
Variables
Condition
Sort
Order
Input Variables:
Name
param1
Edit...

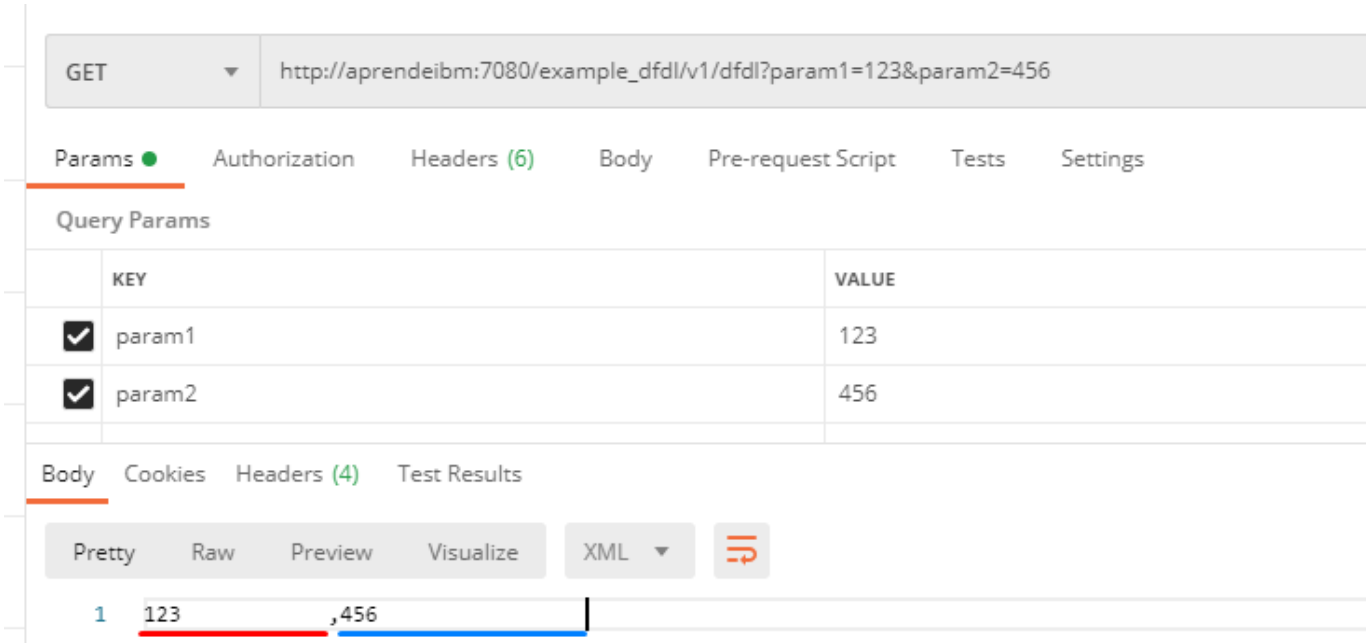

getDfdl_Mapping
Environment _EnvironmentType
<Click to filter...>
Variables [0..1] _EnvironmentVariablesType
any [0..*]
CallableFlow [0..1] _CallableFlowType
any [0..*]
Transferred
Move
Move
Environment _EnvironmentType
<Click to filter...>
Variables [0..1] _EnvironmentVariablesType
CallableFlow [0..1] _CallableFlowType
any [0..*]
Message Assembly message
<Click to filter...>
Properties [0..1] PropertiesType
message [1..1] <Anonymous>
field1 [1..1] string
field2 [1..1] string

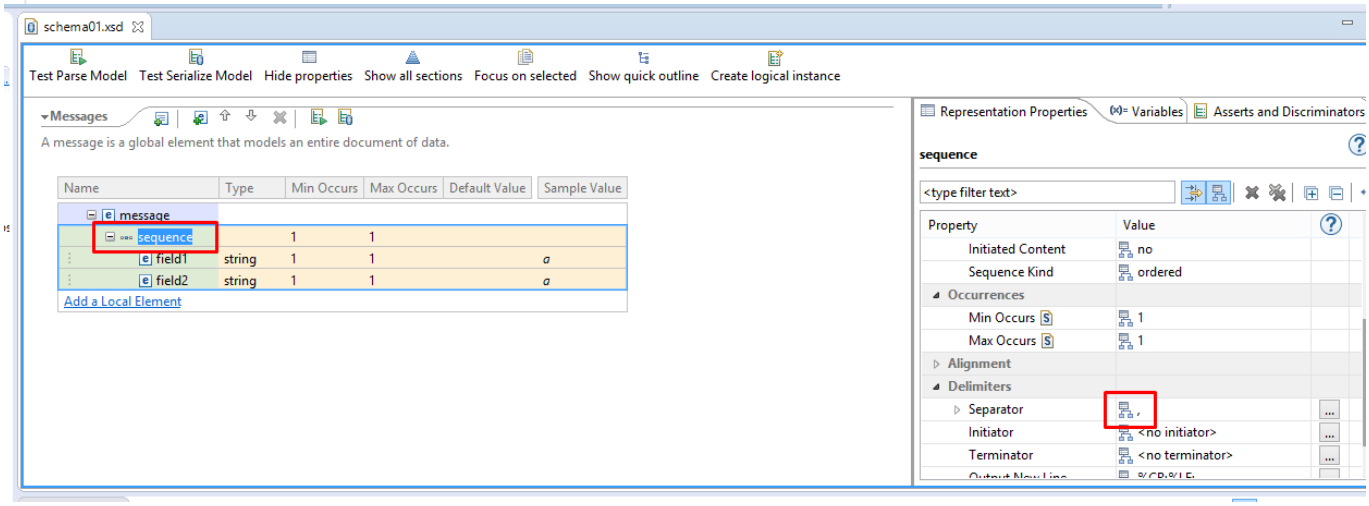

Properties
Problems
Outline
Tasks
Deployment Log
Transform - Move
Cardinality
Variables
Condition
Sort
Input array indexes:
Environment / Variables / any [ 2 ]

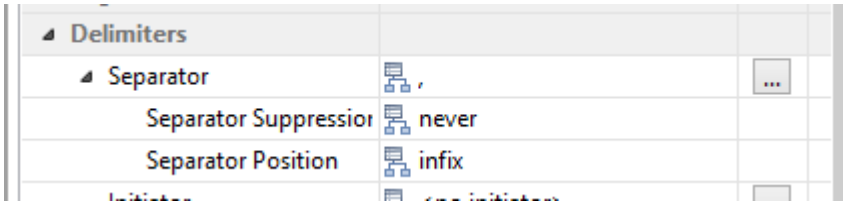

getDfdl_Mapping
Environment _EnvironmentType
<Click to filter...>
Variables [0..1] _EnvironmentVariablesType
any [0..*]
CallableFlow [0..1] _CallableFlowType
any [0..*]
Transferred
Move
Move
Environment _EnvironmentType
<Click to filter...>
Variables [0..1] _EnvironmentVariablesType
CallableFlow [0..1] _CallableFlowType
any [0..*]
Message Assembly message
<Click to filter...>
Properties [0..1] PropertiesType
message [1..1] <Anonymous>
field1 [1..1] string
field2 [1..1] string

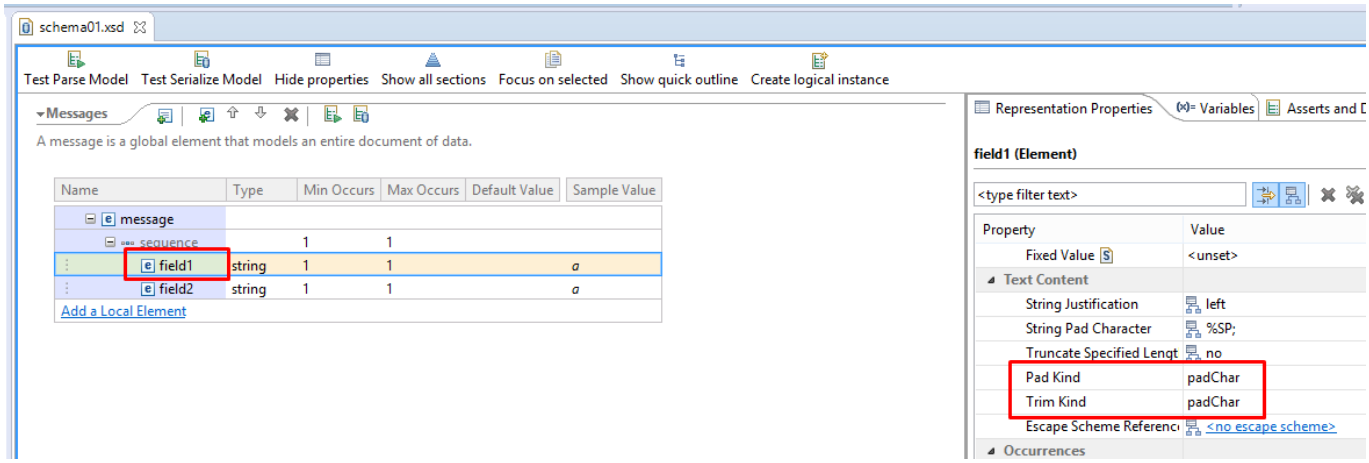

Properties
Problems
Outline
Tasks
Deployment Log
Transform - Move
Cardinality
Variables
Condition
Sort
Input Variables:
Name
param2
Edit...


Example_DFDL
getDfdl.subflow
getDfdl_Compute.esql
Palette
Favorites
IBM MQ
MQTT
Kafka
JMS
HTTP
REST
Web Services
WebSphere Adapters
Routing
.NET
Transformation
Construction
Callable Flow
Cloud Connectors
Flow Exerciser:
Input
Compute
Mapping
MQ Output
MQOutput : MQ Output
Graph
User Defined Properties
Properties
Problems
Outline
Tasks
Deployment Log
MQ Output Node Properties - MQ Output
Description
Basic
MQ Connection
Advanced
Request
Enter the queue name to specify the destination of output messages if you set destination mode to queue name on the 'Advanced' page. Configure connection to the
if needed.
Queue name
QL.IN.AS400
MQ Explorer - Content
Queues
Filter: Standard for Queu
Queue name
QL.01
QL.02
QL.IN.AS400
QL.OUT.AS400
TramaJSON.QL.01
TramaJSON.QL.02
TramaXML.QL.01
TramaXML.QL.02
Message browser
Queue Manager Name: QM_01
Queue Name: QL.IN.AS400
Position
1
2
3
4
5
Scheme: Standard f
Last updated: 19:46:
All available mes
Message 5 - Properties
General
Report
Context
Identifiers
Segmentation
Data
Data
Total length: 36
Data length: 36
Format:
Coded character set identifier: 850
Encoding: 546
Message data: 123 ,456
Message data bytes:
00000 31 32 33 20 20 20 20 20--20 20
00010 34 35 36 20 20 20 20 20--20 20
00020 20 20 20 20 --

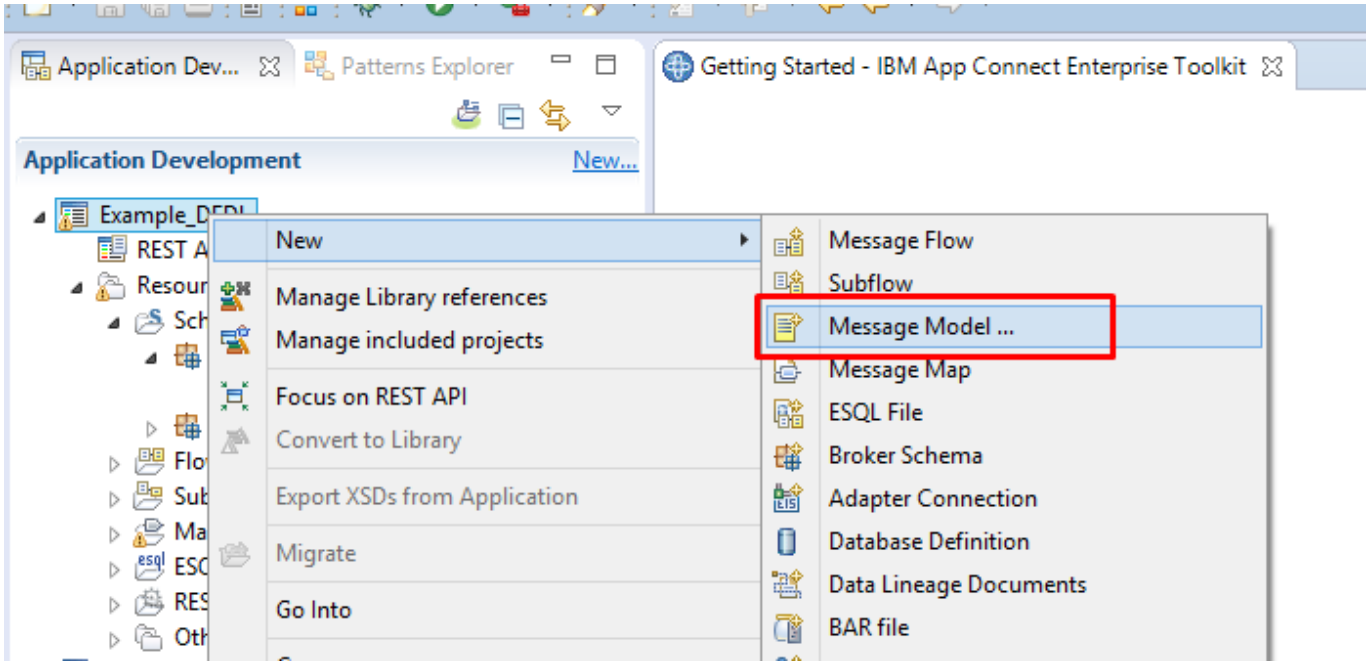

GET
http://aprendeibm:7080/example_dfdl/v1/dfdl?param1=123&param2=456
Params
Authorization
Headers (6)
Body
Pre-request Script
Tests
Settings
Query Params
KEY
VALUE
param1
123
param2
456
Body
Cookies
Headers (4)
Test Results
Pretty
Raw
Preview
Visualize
XML
1
123
,456

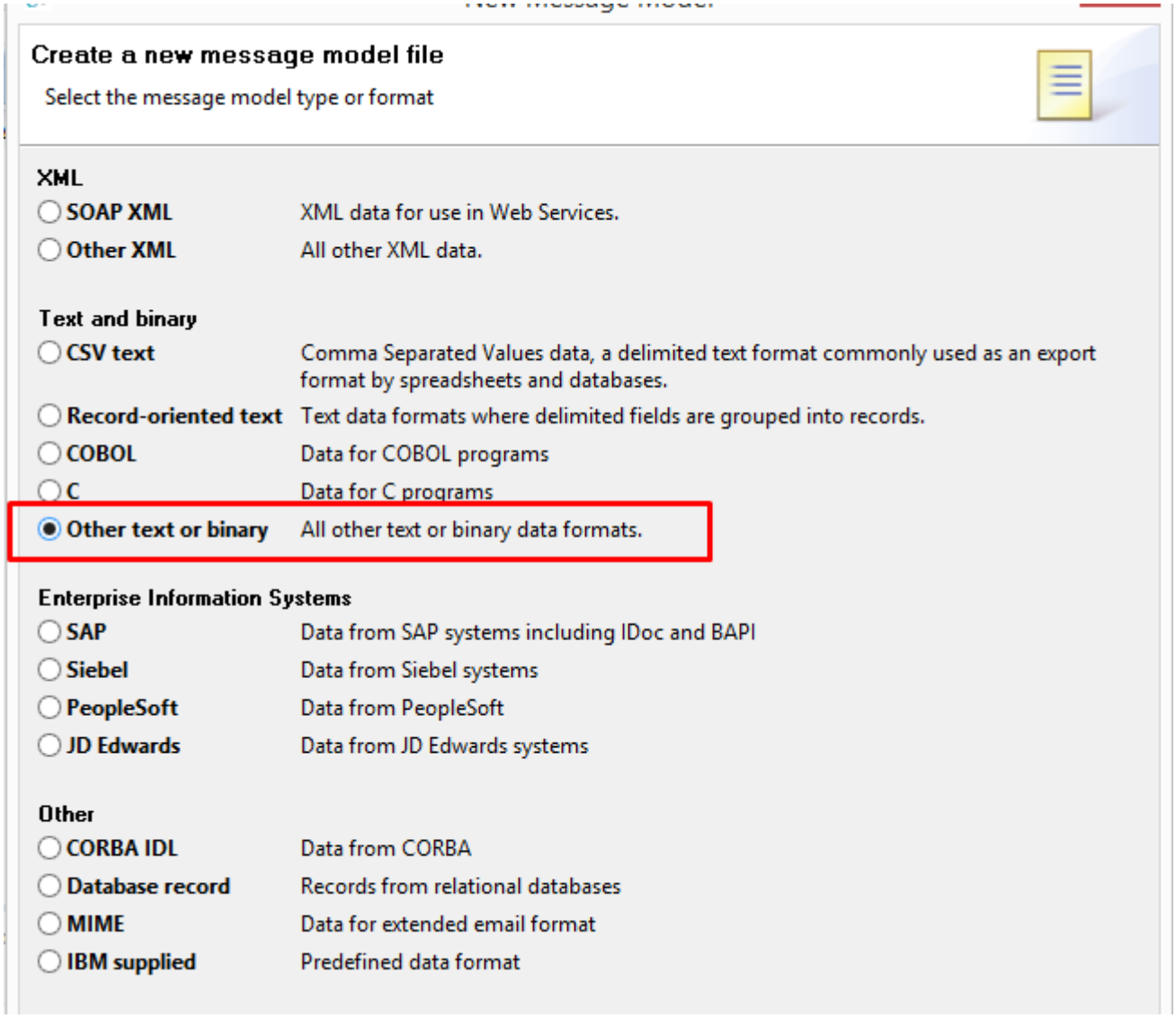

schema01.xsd
Test Parse Model
Test Serialize Model
Hide properties
Show all sections
Focus on selected
Show quick outline
Create logical instance
Messages
A message is a global element that models an entire document of data.
Name
Type
Min Occurs
Max Occurs
Default Value
Sample Value
message
sequence
field1
field2
string
Add a Local Element
Representation Properties
Variables
Asserts and Discriminators
sequence
Property
Value
Initiated Content
no
Sequence Kind
ordered
Occurrences
Min Occurs
Max Occurs
Alignment
Delimiters
Separator
Initiator
<no initiator>
Terminator
<no terminator>

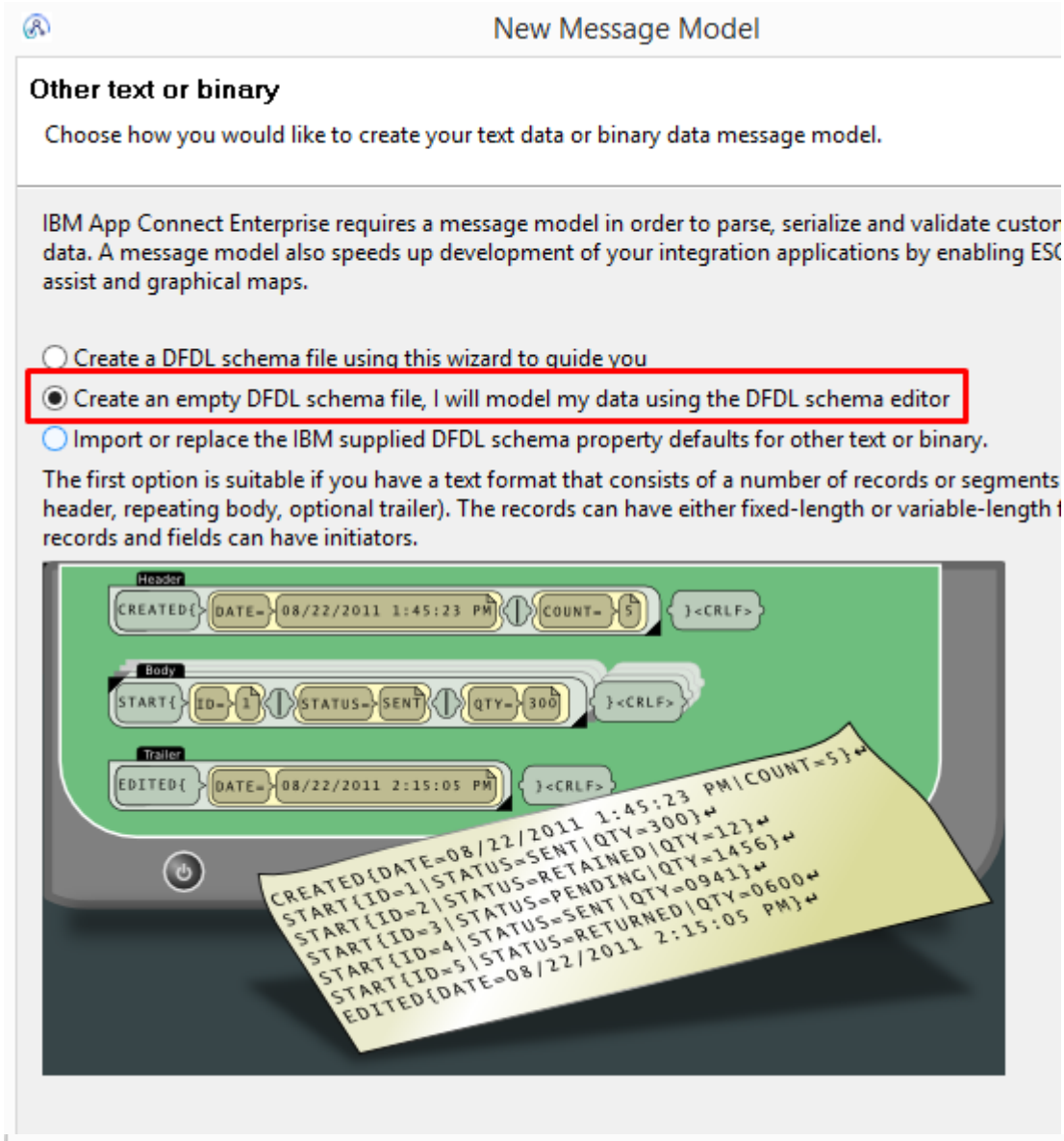

Delimiters
Separator
Separator Suppressior
never
Separator Position
infix

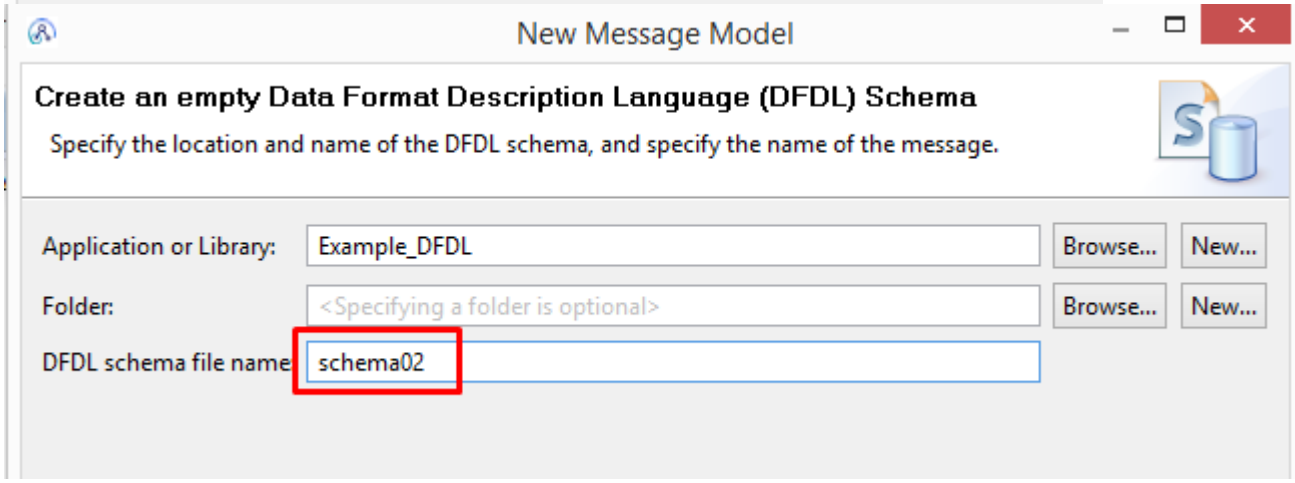


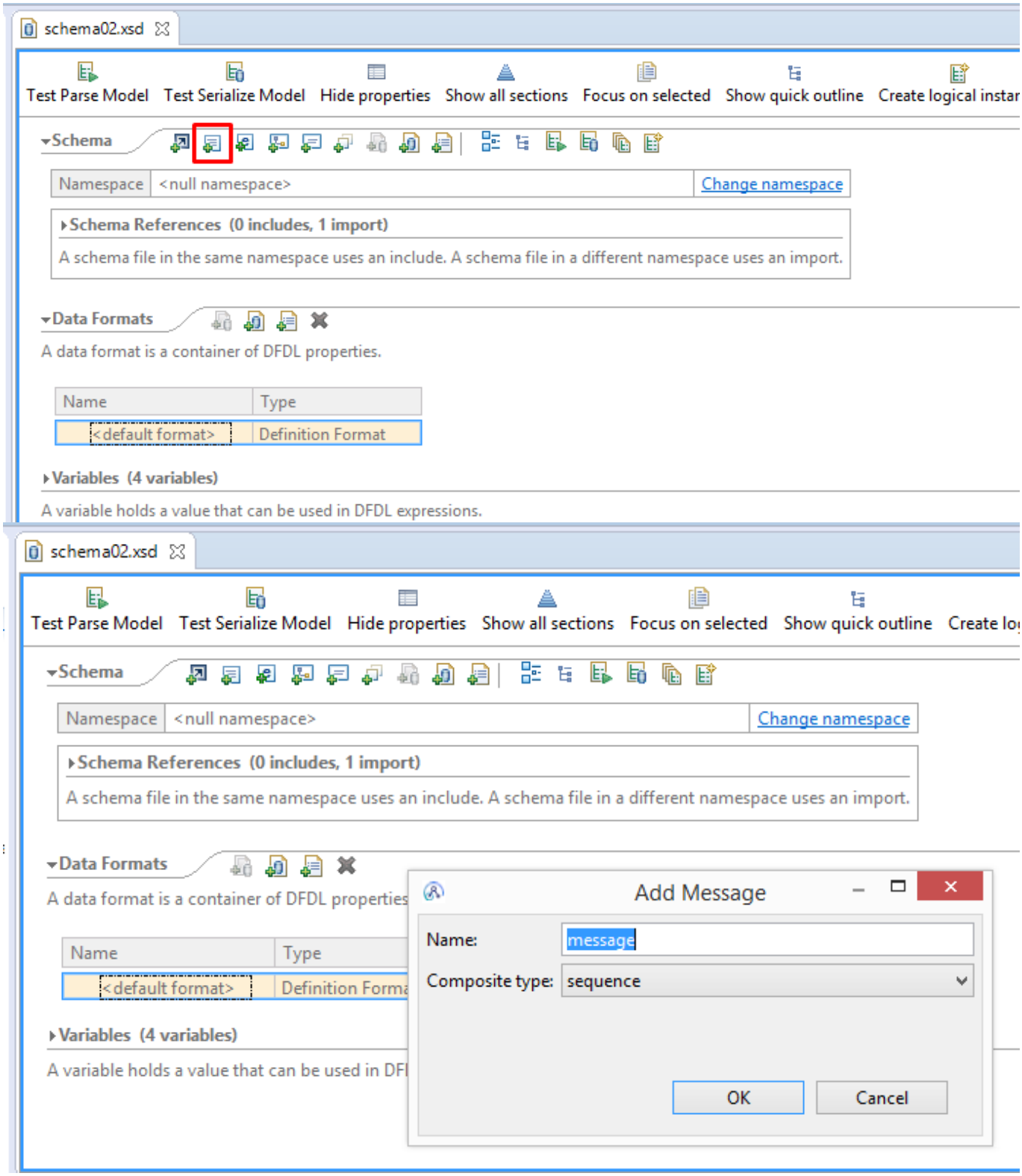

Crear el XSD con la definición a trabajar la trama posicional

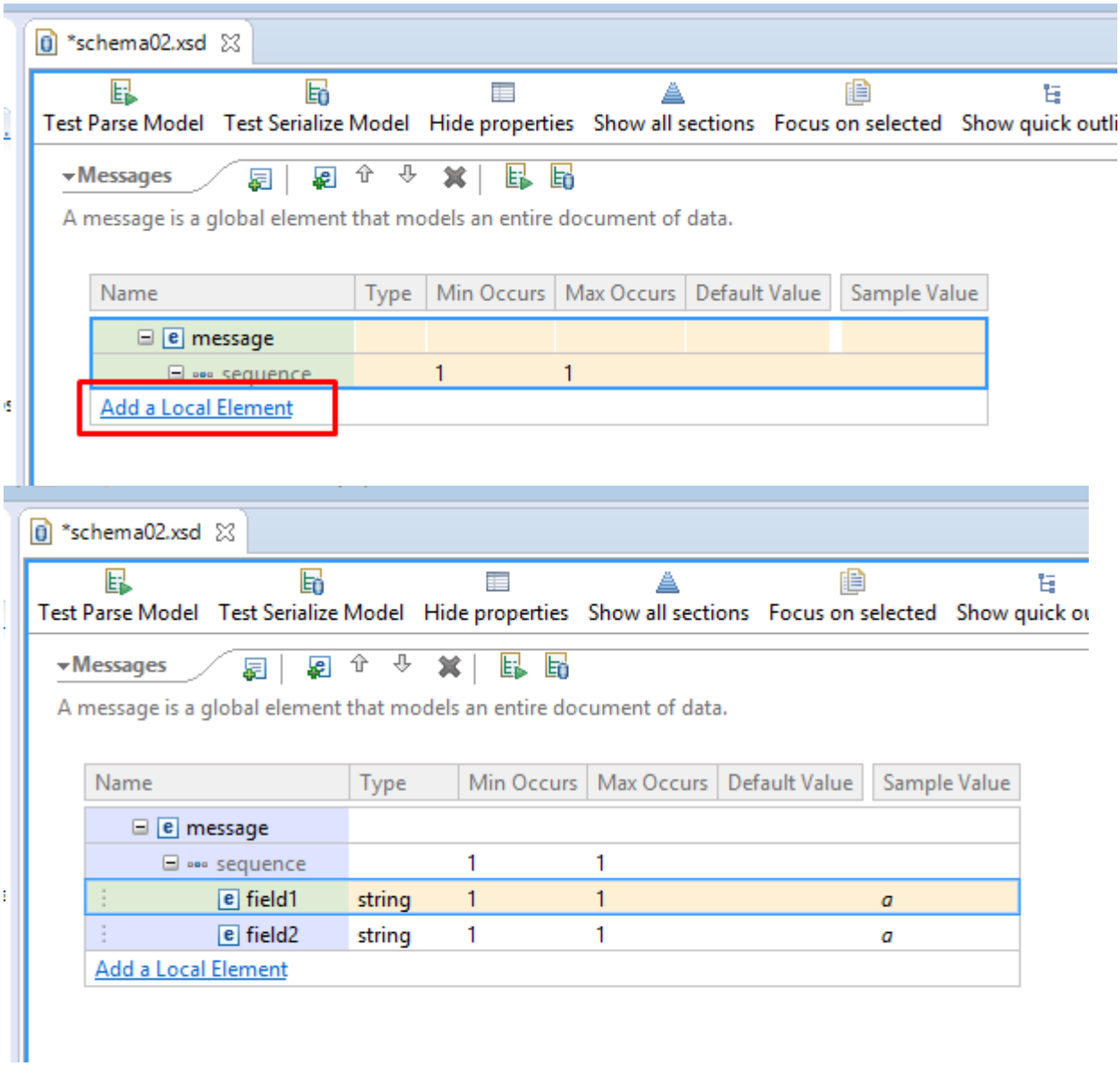



Create a new message model file
Select the message model type or format
XML
SOAP XML
XML data for use in Web Services.
Other XML
All other XML data.
Text and binary
CSV text
Comma Separated Values data, a delimited text format commonly used as an export format by spreadsheets and databases.
Record-oriented text
Text data formats where delimited fields are grouped into records.
COBOL
Data for COBOL programs
C
Data for C programs
Other text or binary
All other text or binary data formats.
Enterprise Information Systems
SAP
Data from SAP systems including IDoc and BAPI
Siebel
Data from Siebel systems
PeopleSoft
Data from PeopleSoft
JD Edwards
Data from JD Edwards systems
Other
CORBA IDL
Data from CORBA
Database record
Records from relational databases
MIME
Data for extended email format
IBM supplied
Predefined data format

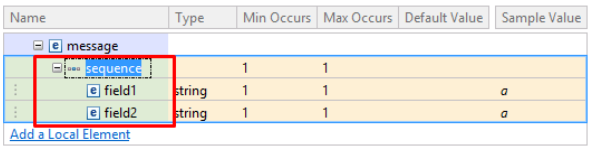

New Message Model
Other text or binary
Choose how you would like to create your text data or binary data message model.
IBM App Connect Enterprise requires a message model in order to parse, serialize and validate custor
data. A message model also speeds up development of your integration applications by enabling ESC
assist and graphical maps.
Create a DFDL schema file using this wizard to guide you
Create an empty DFDL schema file, I will model my data using the DFDL schema editor
Import or replace the IBM supplied DFDL schema property defaults for other text or binary.
The first option is suitable if you have a text format that consists of a number of records or segments
header, repeating body, optional trailer). The records can have either fixed-length or variable-length
records and fields can have initiators.
Header
CREATED{ DATE= 08/22/2011 1:45:23 PM COUNT= 5 }<CRLF>
Body
START{ ID= 1 STATUS= SENT QTY= 300 }<CRLF>
Trailer
EDITED{ DATE= 08/22/2011 2:15:05 PM }<CRLF>
CREATED{DATE=08/22/2011 1:45:23 PM|COUNT=5}
START{ID=1|STATUS=SENT|QTY=300}
START{ID=2|STATUS=RETAINED|QTY=12}
START{ID=3|STATUS=PENDING|QTY=1456}
START{ID=4|STATUS=SENT|QTY=0941}
START{ID=5|STATUS=RETURNED|QTY=0600}
EDITED{DATE=08/22/2011 2:15:05 PM}

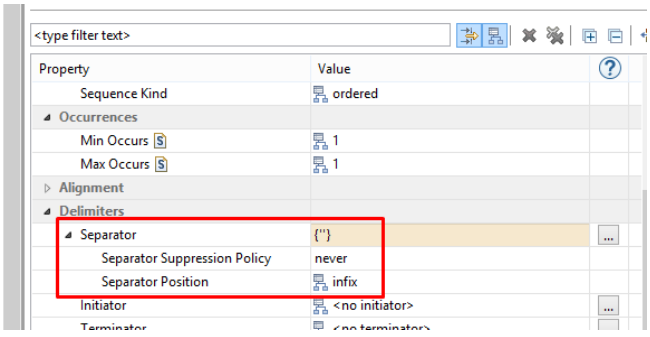

New Message Model
Create an empty Data Format Description Language (DFDL) Schema
Specify the location and name of the DFDL schema, and specify the name of the message.
Application or Library: Example_DFDL
Browse...
New...
Folder: <Specifying a folder is optional>
Browse...
New...
DFDL schema file name: schema02

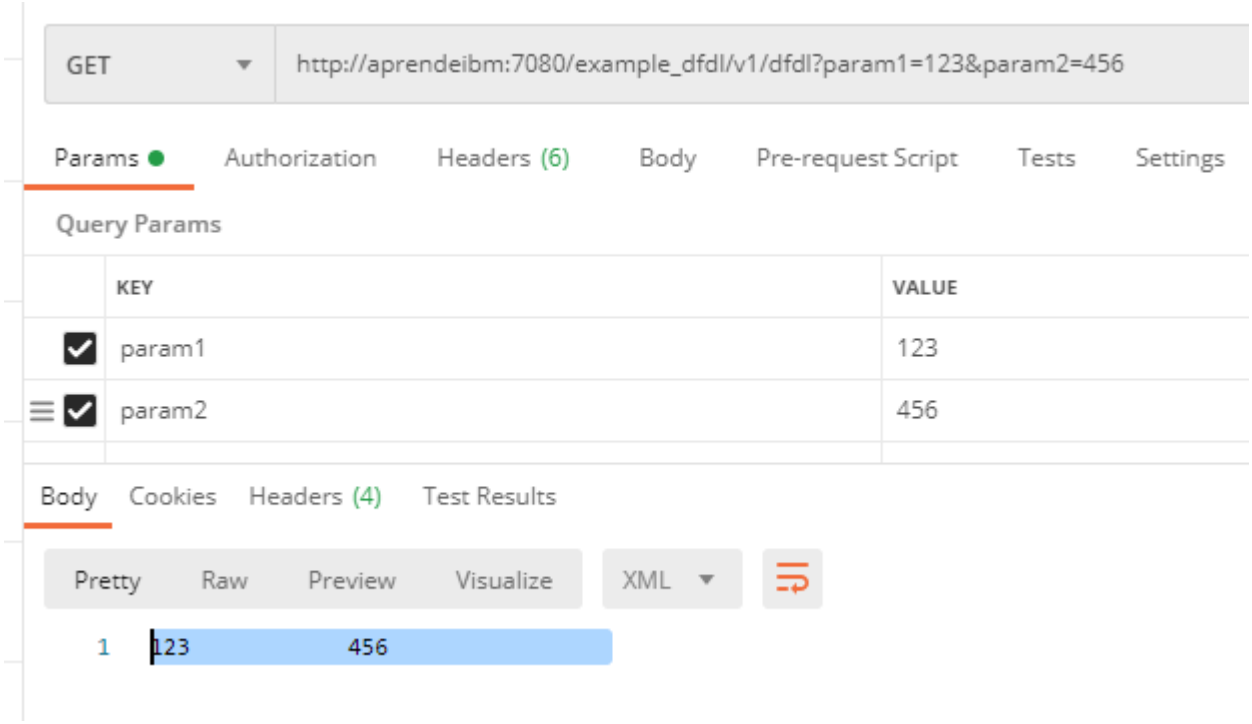

schema02.xsd
Test Parse Model
Test Serialize Model
Hide properties
Show all sections
Focus on selected
Show quick outline
Create logical instar
Schema
Namespace
<null namespace>
Change namespace
Schema References (0 includes, 1 import)
A schema file in the same namespace uses an include. A schema file in a different namespace uses an import.
Data Formats
A data format is a container of DFDL properties.
Name
Type
<default format>
Definition Format
Variables (4 variables)
A variable holds a value that can be used in DFDL expressions.
schema02.xsd
Test Parse Model
Test Serialize Model
Hide properties
Show all sections
Focus on selected
Show quick outline
Schema
Namespace
<null namespace>
Change namespace
Schema References (0 includes, 1 import)
A schema file in the same namespace uses an include. A schema file in a different namespace uses an import.
Data Formats
A data format is a container of DFDL properties
Name
Type
<default format>
Variables (4 variables)
Add Message
Name:
message
Composite type:
sequence
OK
Cancel




En caso que se requiera que el separador no sea coma y los valores esten pegados, setear






## ArquitecturaIBM Consulting