z/TPF Descriptor Definition Projects

TPF Toolkit support for Business events and DFDL

Matt Gritter
TPF Toolkit Technical Lead

IBM **z/TPF**

April 12, 2016

Overview

Views and Wizards

DFDL editor

Q&A

Agenda

Overview

- A few things that should be understood before using the descriptor definition project support in TPF Toolkit
  - Common Deployment Descriptors
  - Business events
  - Data Format Description Language (DFDL)

Common Deployment Descriptors

Business events

DFDL

Deployment Descriptors

- XML file that describes the capabilities and options for a specific function or component
- Unique file extension to identify the function or component
- Refer to the Knowledge Center for more detailed information

Common Deployment Descriptors

Business events

DFDL

Business events

- Two types
  - Signal – triggered by application call
  - Data – triggered by data modification
    - z/TPF or z/TPFDF file
- Knowledge Center

Common Deployment Descriptors

Business events

DFDL

DFDL

- Open standard
- Describes text and binary data formats
- Data can be presented as information set
- Annotated XML schema as logical model
- Transform data from native to XML
- IBM developerWorks
- Additional presentations at TPFUG

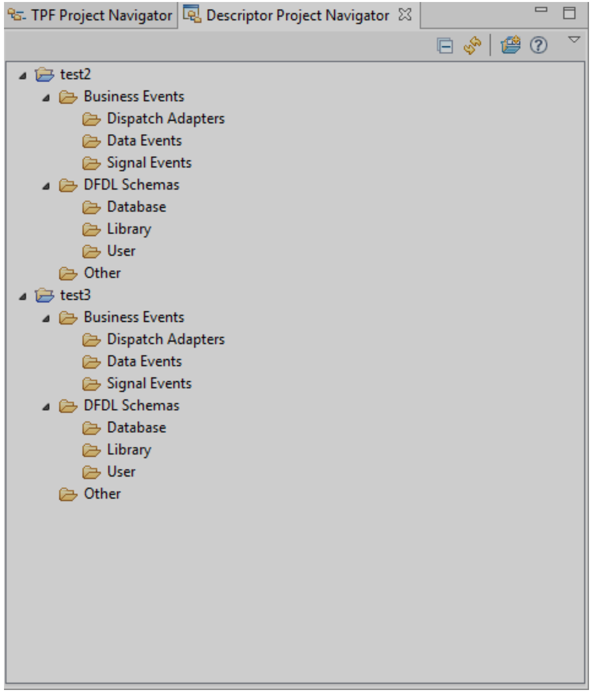
Descriptor Definition Projects

- This iteration provides a view and wizards that help organize, create, and load business event artifacts
- Future - expand tooling to support similar artifact based TPF components

Views and Wizards

- Views
  - Descriptor Project Navigator
- Wizards
  - New Descriptor Project
  - New Dispatch Adapter
  - New Data Event Message
  - New Data Event Specification
  - New z/TPF File Collection Descriptor
  - New z/TPF File Event Data
  - New Signal Event Message
  - New Signal Event Specification
  - New Library Data Definition
  - New User Data Definition

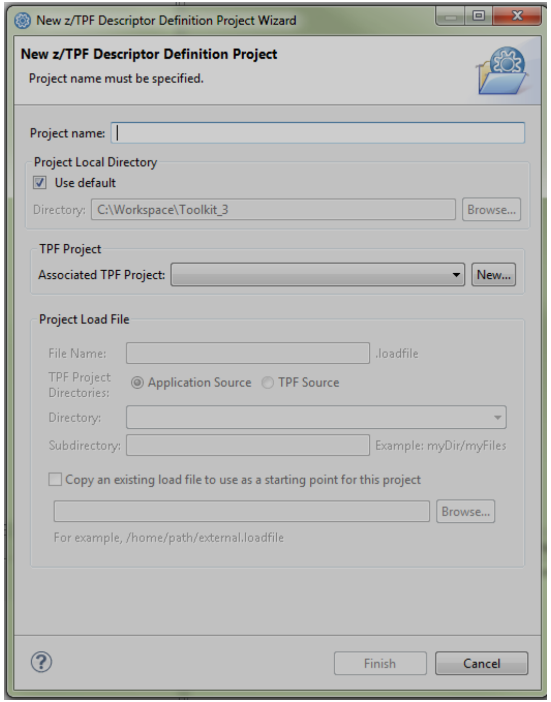
Descriptor Project Navigator



```
◢ test2
    ◢ Business Events
        Dispatch Adapters
        Data Events
        Signal Events
    ◢ DFDL Schemas
        Database
        Library
        User
    Other
◢ test3
    ◢ Business Events
        Dispatch Adapters
        Data Events
        Signal Events
    ◢ DFDL Schemas
        Database
        Library
        User
    Other
```

- A Descriptor project is a collection of virtual folders that organize different types of files
- Each Descriptor project is associated with a TPF Project
- Sub-folders provide actions to create new files using wizards
- Files can be edited using the same wizards

New Descriptor Project Wizard



New z/TPF Descriptor Definition Project Wizard

New z/TPF Descriptor Definition Project

Project name must be specified.

Project name:

Project Local Directory

☑ Use default

Directory: C:\Workspace\Toolkit_3

TPF Project

Associated TPF Project:

Project Load File

File Name: .loadfile

TPF Project Directories: ● Application Source ○ TPF Source

Directory:

Subdirectory: Example: myDir/myFiles

□ Copy an existing load file to use as a starting point for this project

For example, /home/path/external.loadfile

- Associated TPF Project provides access to maketpf variables
- Descriptor project contents are tracked via a load file
  - Call loadtpf using load file from Descriptor project directly
  - Or add descriptor load file to TPF Project

Dispatch Adapter

Data Events

Signal Events

Dispatch Adapter

- Completing the wizard creates a deployment descriptor file
- A dispatch adapter specifies the data format and how the data is transmitted

Dispatch Adapter

Data Events

Signal Events

Data Events

- Create DFDL schema for event data
- Define the event message format
- Create event specification to define event
- Load and activate project loadset

- Can also define events for file collections

Dispatch Adapter

Data Events

Signal Events

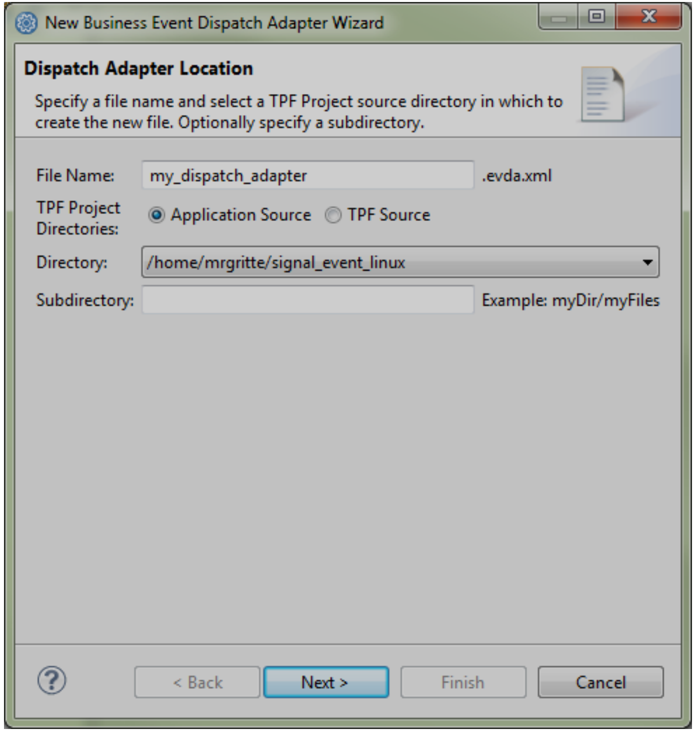
Signal Events

- Define event message format
- Create event specification to define event
- Load and activate project loadset

DFDL editor

- Common DFDL editor provided by the IBM DFDL team
- The DFDL Test perspective is shipped along with the editor

- As an example, step through the process of creating a simple signal event.
- Created a new descriptor project: my_signal_event



Dispatch Adapter Location

Specify a file name and select a TPF Project source directory in which to create the new file. Optionally specify a subdirectory.

File Name: my_dispatch_adapter .evda.xml

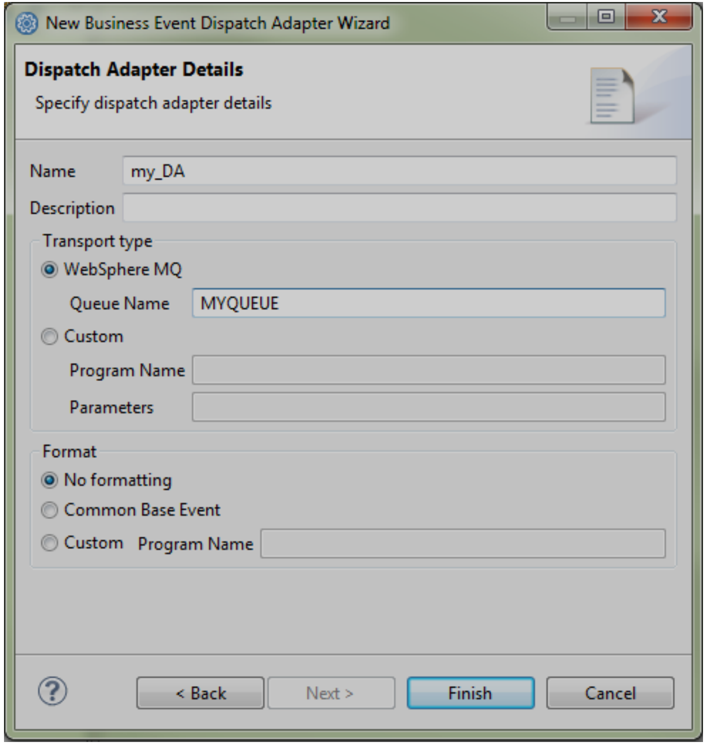
TPF Project Directories: ● Application Source ○ TPF Source

Directory: /home/mrgritte/signal_event_linux

Subdirectory: Example: myDir/myFiles

- Create a new dispatch adapter using the wizard

- File Name: my_dispatch_adapter



Dispatch Adapter Details

Specify dispatch adapter details

Name: my_DA

Description:

Transport type

● WebSphere MQ

Queue Name: MYQUEUE

○ Custom

Program Name:

Parameters:

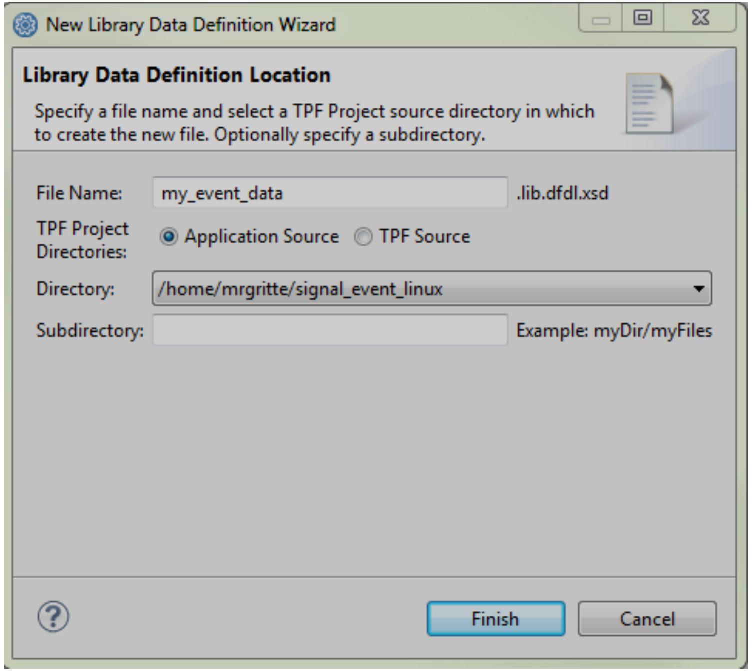
Format

● No formatting

○ Common Base Event

○ Custom Program Name:

- Name: my_DA
- WebSphere MQ Queue Name: MYQUEUE
- No formatting



New Library Data Definition Wizard

Library Data Definition Location

Specify a file name and select a TPF Project source directory in which to create the new file. Optionally specify a subdirectory.

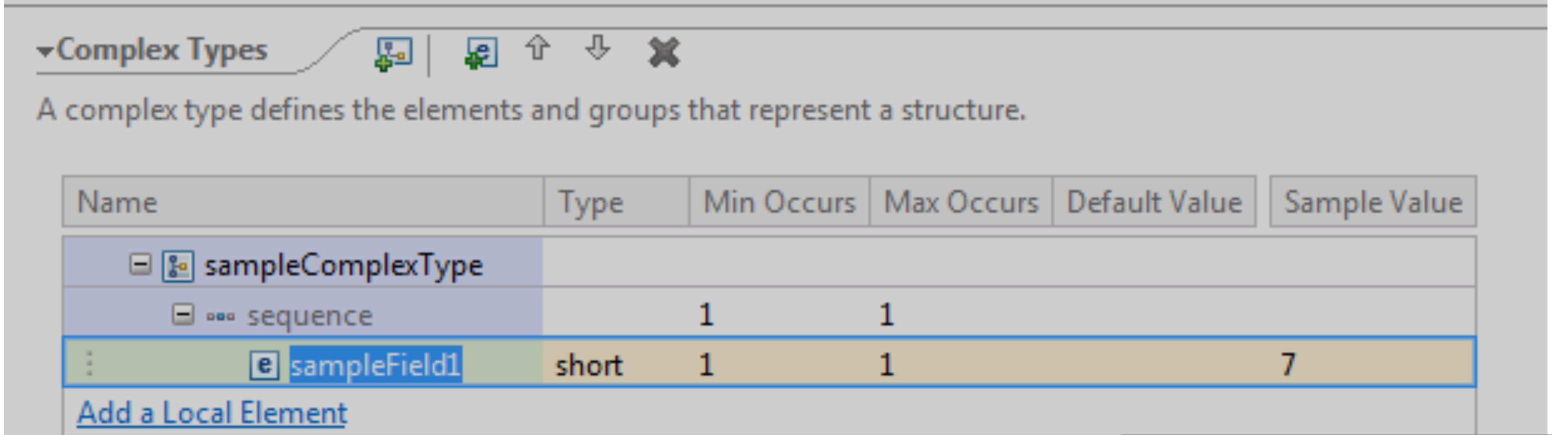
File Name: my_event_data .lib.dfdl.xsd

TPF Project Directories: ● Application Source ○ TPF Source

Directory: /home/mrgritte/signal_event_linux

Subdirectory: Example: myDir/myFiles

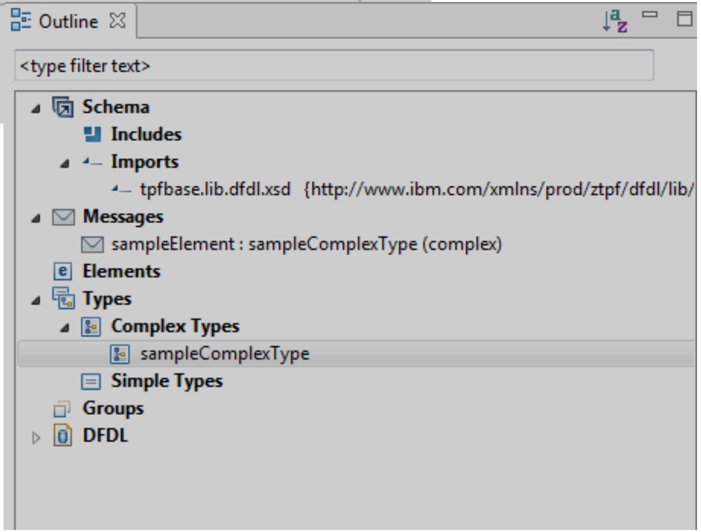
- Create a DFDL schema file to describe event data
- Use the Library Data Definition wizard to create
- File Name: my_event_data
- my_event_data.lib.dfdl.xsd is created with the complex type sampleComplexType defined.
- Rename to my_complex_type
- Add some additional fields to make things interesting



Complex Types

A complex type defines the elements and groups that represent a structure.

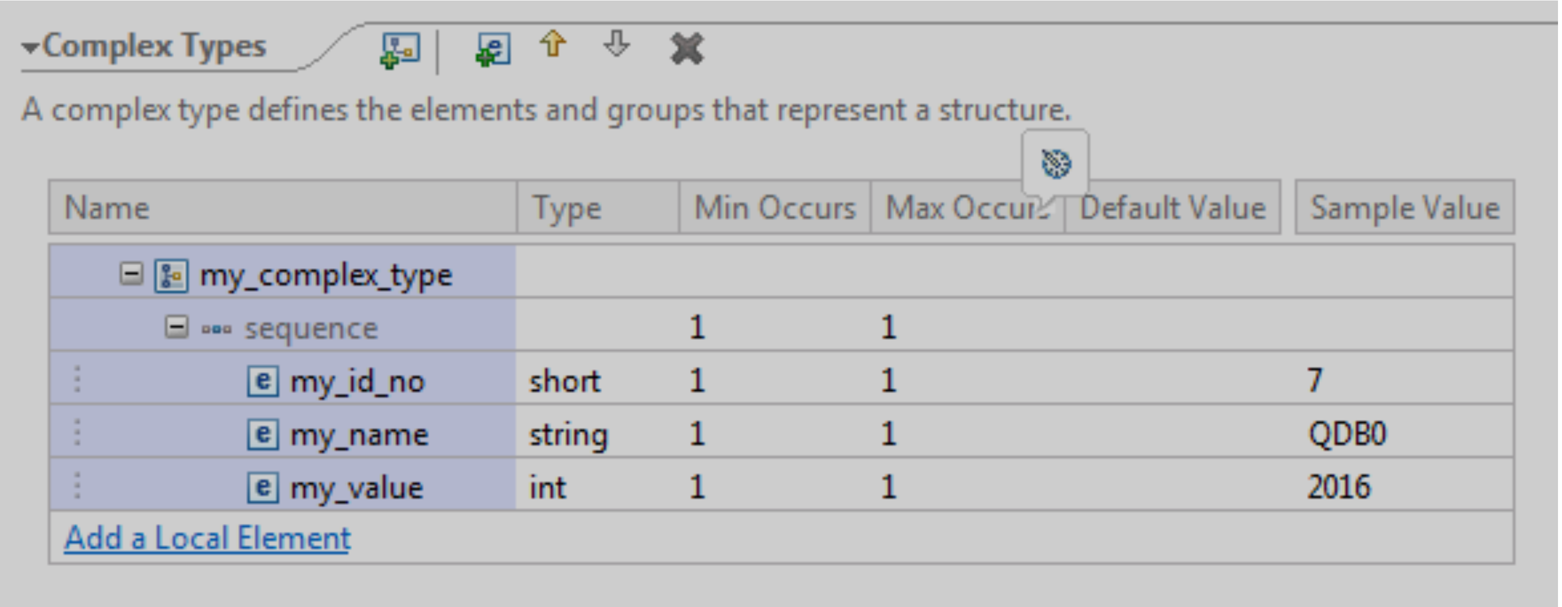
Name	Type	Min Occurs	Max Occurs	Default Value	Sample Value
⊟ sampleComplexType					
⊟ sequence		1	1		
e sampleField1	short	1	1		7
Add a Local Element					



```
◢ Schema
    Includes
  ◢ Imports
      tpfbase.lib.dfdl.xsd  {http://www.ibm.com/xmlns/prod/ztpf/dfdl/lib/
◢ Messages
    sampleElement : sampleComplexType (complex)
  Elements
◢ Types
  ◢ Complex Types
      sampleComplexType
    Simple Types
  Groups
▷ DFDL
```

Before

DFDL editor



A complex type defines the elements and groups that represent a structure.

Name	Type	Min Occurs	Max Occurs	Default Value	Sample Value
my_complex_type					
sequence		1	1		
my_id_no	short	1	1		7
my_name	string	1	1		QDB0
my_value	int	1	1		2016

After

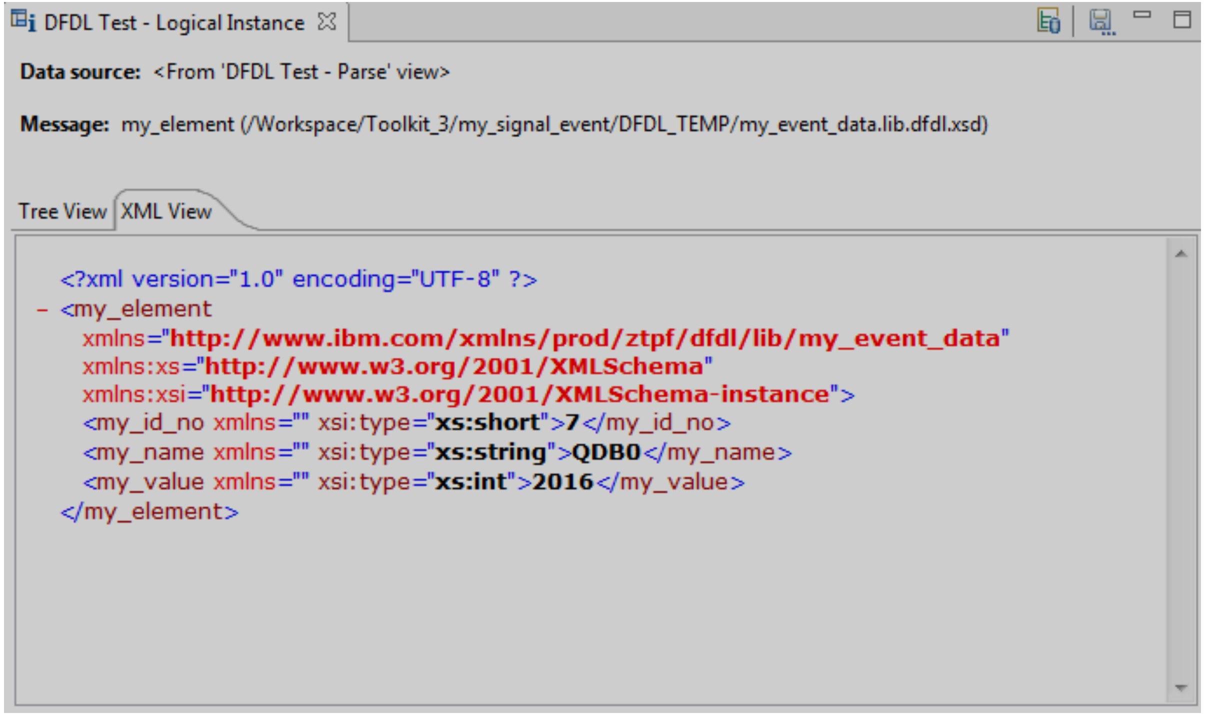
Added fields with sample values

- my_id_no (2 bytes)
- my_name (4 bytes)
- my_value (4 bytes)

DFDL editor

After editing the DFDL schema, create a logical instance of the event data

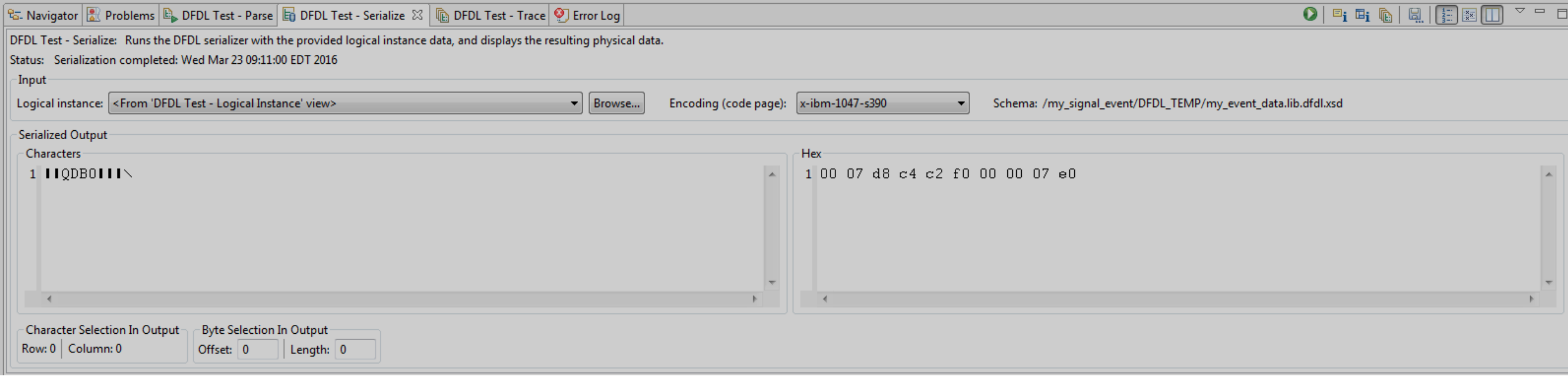
XML document that adheres to the schema definition created and populated with the sample data provided



Data source: <From 'DFDL Test - Parse' view>

Message: my_element (/Workspace/Toolkit_3/my_signal_event/DFDL_TEMP/my_event_data.lib.dfdl.xsd)

```
<?xml version="1.0" encoding="UTF-8" ?>
- <my_element
    xmlns="http://www.ibm.com/xmlns/prod/ztpf/dfdl/lib/my_event_data"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <my_id_no xmlns="" xsi:type="xs:short">7</my_id_no>
    <my_name xmlns="" xsi:type="xs:string">QDB0</my_name>
    <my_value xmlns="" xsi:type="xs:int">2016</my_value>
  </my_element>
```



DFDL Test - Serialize: Runs the DFDL serializer with the provided logical instance data, and displays the resulting physical data.

Status: Serialization completed: Wed Mar 23 09:11:00 EDT 2016

Input

Logical instance: <From 'DFDL Test - Logical Instance' view> ▾ Browse... Encoding (code page): x-ibm-1047-s390 ▾ Schema: /my_signal_event/DFDL_TEMP/my_event_data.lib.dfdl.xsd

Serialized Output

Characters

```
1 ▌▌QDB0▌▌▌\
```

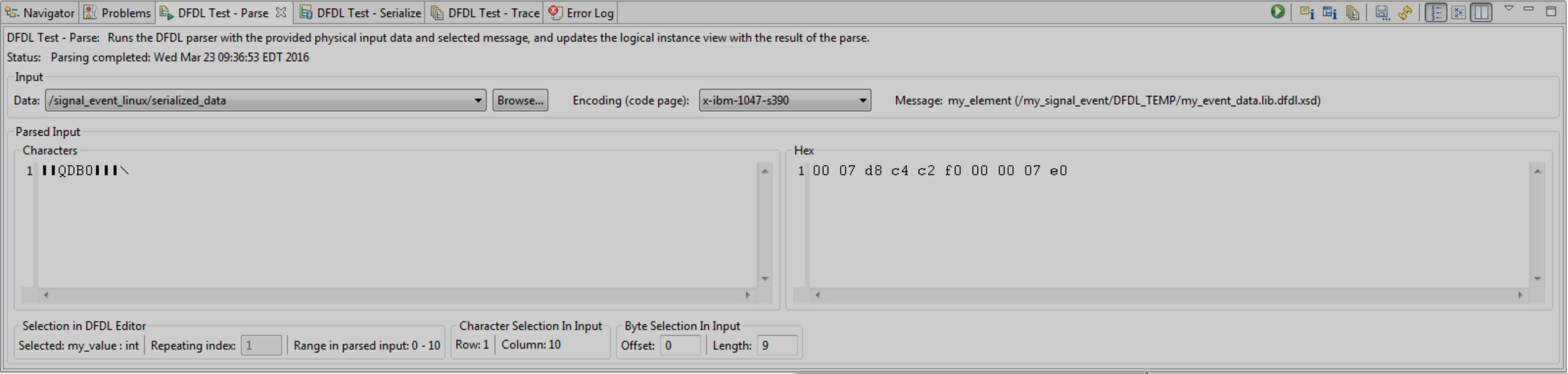
Hex

```
1 00 07 d8 c4 c2 f0 00 00 07 e0
```

Character Selection In Output: Row: 0 Column: 0

Byte Selection In Output: Offset: 0 Length: 0

- Take the logical instance and serialize it into binary data.
- Save the binary data out to a file named serialized_data



DFDL Test - Parse: Runs the DFDL parser with the provided physical input data and selected message, and updates the logical instance view with the result of the parse.

Status: Parsing completed: Wed Mar 23 09:36:53 EDT 2016

Input

Data: /signal_event_linux/serialized_data Browse... Encoding (code page): x-ibm-1047-s390 Message: my_element (/my_signal_event/DFDL_TEMP/my_event_data.lib.dfdl.xsd)

Parsed Input

Characters

```
1 ▮▮QDB0▮▮▮\
```

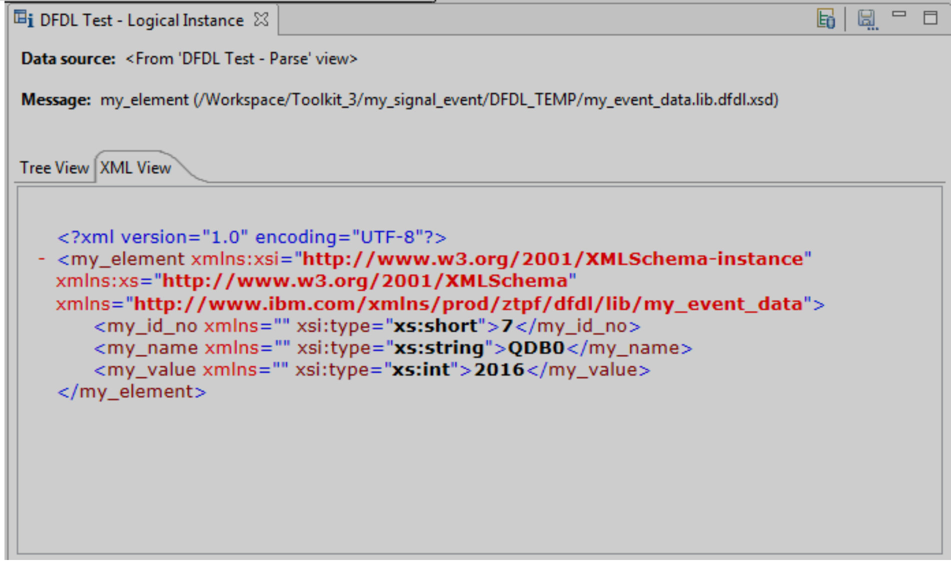
Hex

```
1 00 07 d8 c4 c2 f0 00 00 07 e0
```

Selection in DFDL Editor: Selected: my_value : int | Repeating index: 1 | Range in parsed input: 0 - 10

Character Selection In Input: Row: 1 | Column: 10

Byte Selection In Input: Offset: 0 | Length: 9



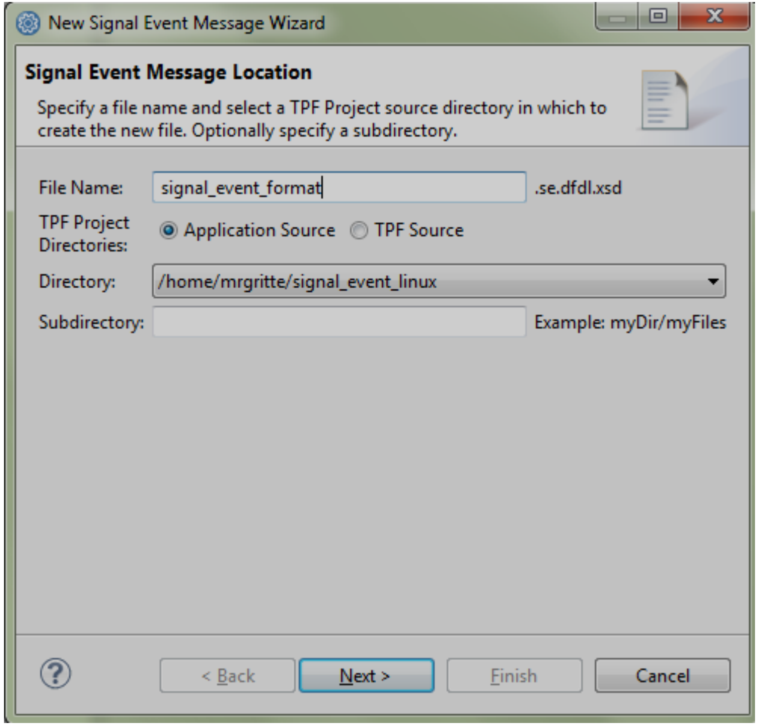
Data source: <From 'DFDL Test - Parse' view>

Message: my_element (/Workspace/Toolkit_3/my_signal_event/DFDL_TEMP/my_event_data.lib.dfdl.xsd)

```
<?xml version="1.0" encoding="UTF-8"?>
- <my_element xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns="http://www.ibm.com/xmlns/prod/ztpf/dfdl/lib/my_event_data">
     <my_id_no xmlns="" xsi:type="xs:short">7</my_id_no>
     <my_name xmlns="" xsi:type="xs:string">QDB0</my_name>
     <my_value xmlns="" xsi:type="xs:int">2016</my_value>
  </my_element>
```

- Parse the binary data in the saved file

- Check the results



Signal Event Message Location

Specify a file name and select a TPF Project source directory in which to create the new file. Optionally specify a subdirectory.

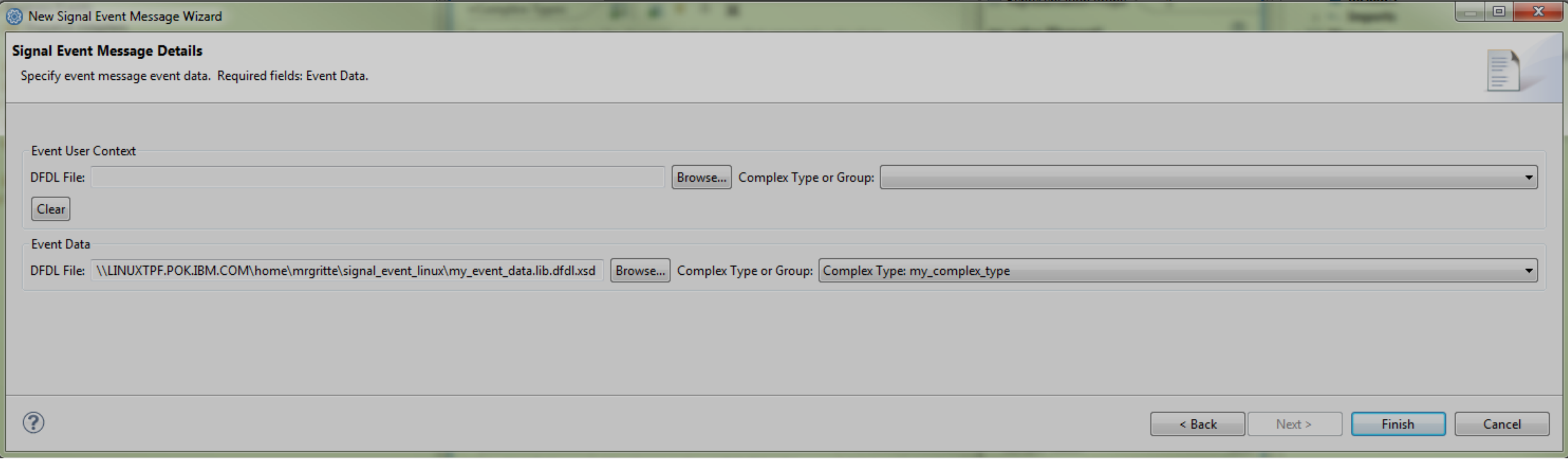
File Name: signal_event_format .se.dfdl.xsd

TPF Project Directories: ● Application Source ○ TPF Source

Directory: /home/mrgritte/signal_event_linux

Subdirectory: Example: myDir/myFiles

- Create new signal event message format using the wizard
- File Name: signal_event_format



Signal Event Message Details

Specify event message event data. Required fields: Event Data.

Event User Context

DFDL File:

Complex Type or Group:

Event Data

DFDL File: \\LINUXTPF.POK.IBM.COM\home\mrgritte\signal_event_linux\my_event_data.lib.dfdl.xsd

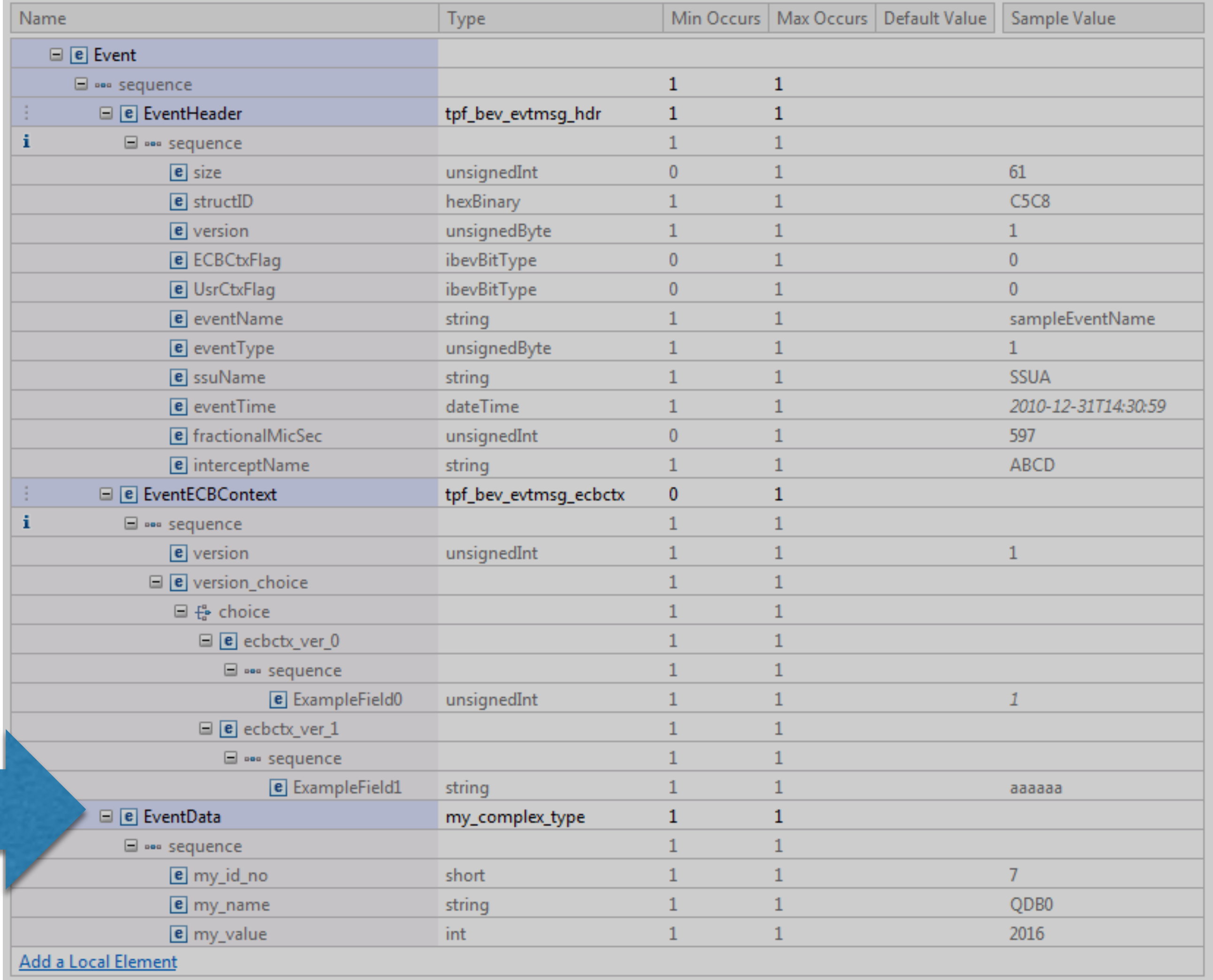
Complex Type or Group: Complex Type: my_complex_type

- Select my_event_data file and select my_complex_type for Event Data

DFDL editor

Event content in signal_event_format

Logical instance
Serialize
Parse



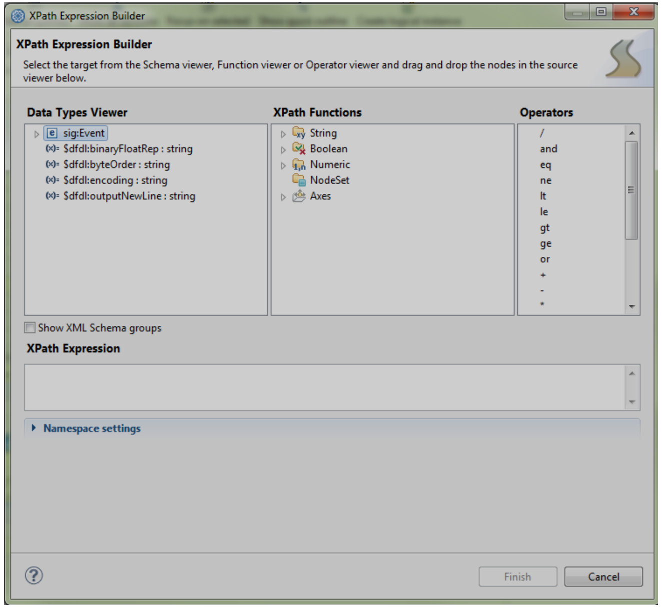
Name	Type	Min Occurs	Max Occurs	Default Value	Sample Value
Event					
sequence		1	1		
EventHeader	tpf_bev_evtmsg_hdr	1	1		
sequence		1	1		
size	unsignedInt	0	1		61
structID	hexBinary	1	1		C5C8
version	unsignedByte	1	1		1
ECBCtxFlag	ibevBitType	0	1		0
UsrCtxFlag	ibevBitType	0	1		0
eventName	string	1	1		sampleEventName
eventType	unsignedByte	1	1		1
ssuName	string	1	1		SSUA
eventTime	dateTime	1	1		*2010-12-31T14:30:59*
fractionalMicSec	unsignedInt	0	1		597
interceptName	string	1	1		ABCD
EventECBContext	tpf_bev_evtmsg_ecbctx	0	1		
sequence		1	1		
version	unsignedInt	1	1		1
version_choice		1	1		
choice		1	1		
ecbctx_ver_0		1	1		
sequence		1	1		
ExampleField0	unsignedInt	1	1		*1*
ecbctx_ver_1		1	1		
sequence		1	1		
ExampleField1	string	1	1		aaaaaa
EventData	my_complex_type	1	1		
sequence		1	1		
my_id_no	short	1	1		7
my_name	string	1	1		QDB0
my_value	int	1	1		2016

Add a Local Element

my_event_data

DFDL editor

- Can add asserts or discriminators to elements to assist in more complex parsing scenarios
- XPath expression builder provided



XPath Expression Builder

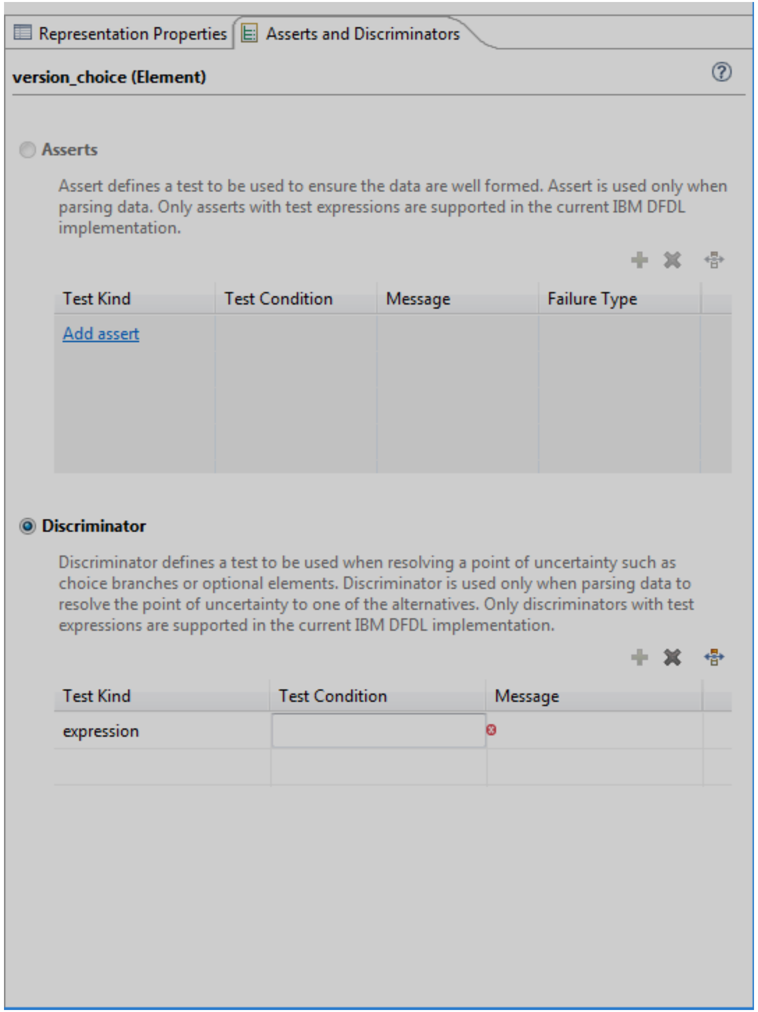
Select the target from the Schema viewer, Function viewer or Operator viewer and drag and drop the nodes in the source viewer below.

Data Types Viewer	XPath Functions	Operators
sig:Event	String	/
$dfdl:binaryFloatRep : string	Boolean	and
$dfdl:byteOrder : string	Numeric	eq
$dfdl:encoding : string	NodeSet	ne
$dfdl:outputNewLine : string	Axes	lt
		le
		gt
		ge
		or
		+
		-
		*

□ Show XML Schema groups

XPath Expression

Namespace settings



version_choice (Element)

○ **Asserts**

Assert defines a test to be used to ensure the data are well formed. Assert is used only when parsing data. Only asserts with test expressions are supported in the current IBM DFDL implementation.

Test Kind	Test Condition	Message	Failure Type
Add assert			

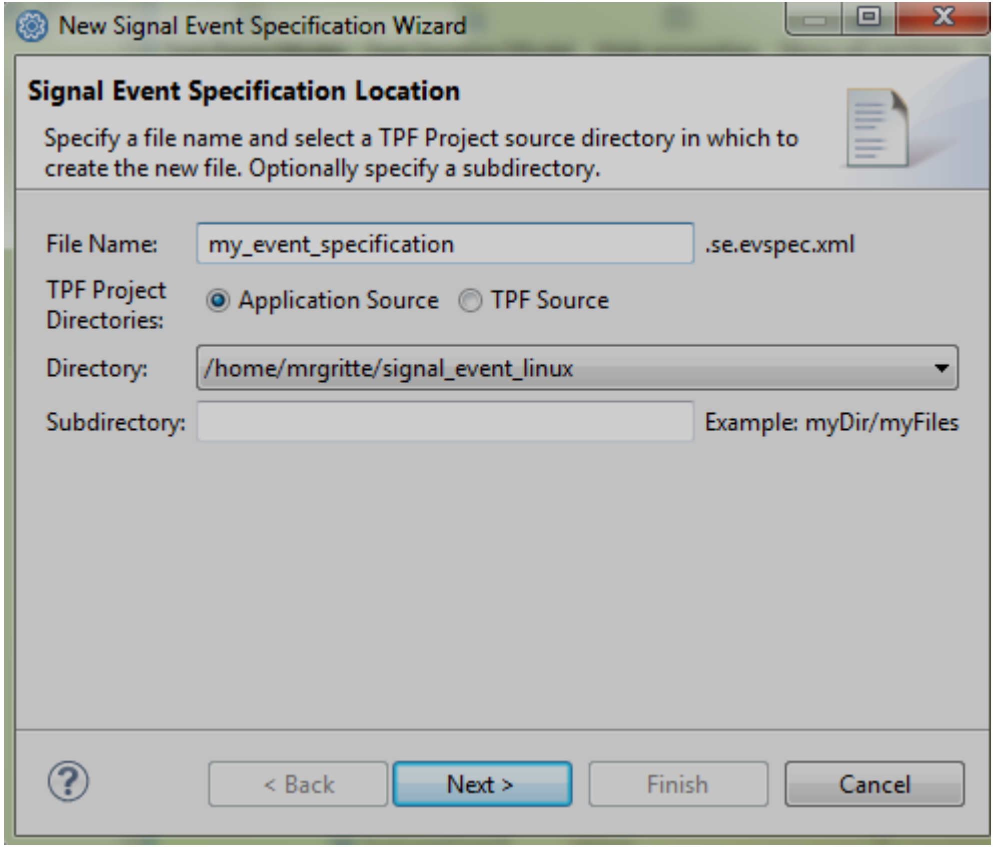
● **Discriminator**

Discriminator defines a test to be used when resolving a point of uncertainty such as choice branches or optional elements. Discriminator is used only when parsing data to resolve the point of uncertainty to one of the alternatives. Only discriminators with test expressions are supported in the current IBM DFDL implementation.

Test Kind	Test Condition	Message
expression		

DFDL editor

- Finally, run the Signal Event Specification Wizard
- File Name: my_event_specification



New Signal Event Specification Wizard

Signal Event Specification Location

Specify a file name and select a TPF Project source directory in which to create the new file. Optionally specify a subdirectory.

File Name: my_event_specification .se.evspec.xml

TPF Project Directories: ● Application Source ○ TPF Source

Directory: /home/mrgritte/signal_event_linux

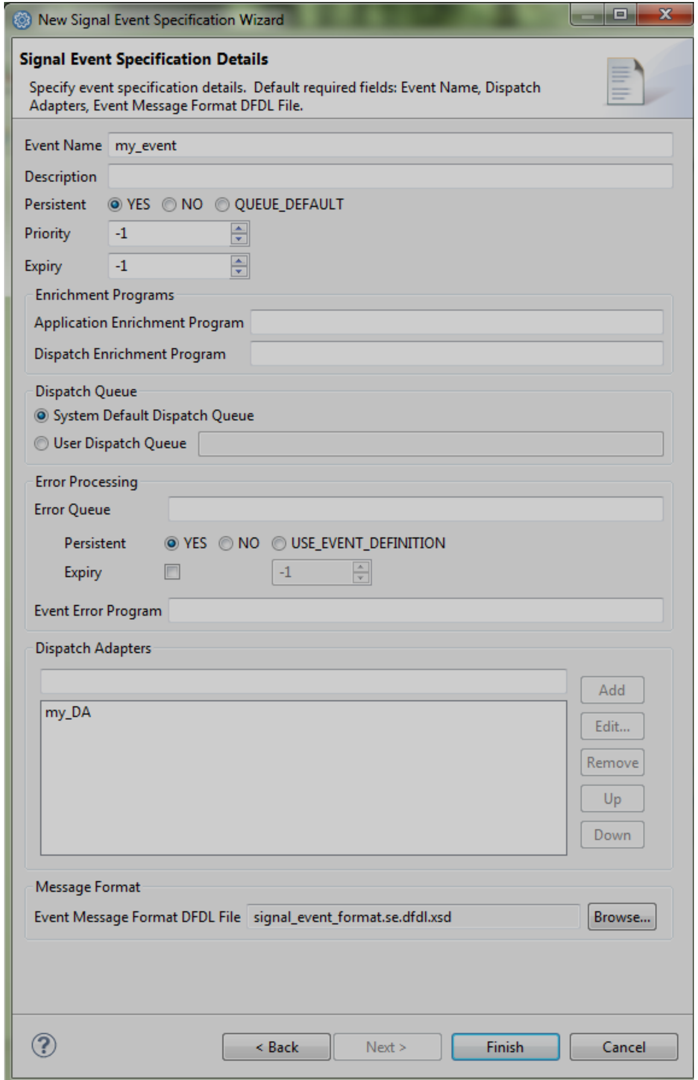
Subdirectory: Example: myDir/myFiles

DFDL editor

- Event Name: my_event

- Dispatch Adapter: my_DA
  Name value from dispatch adapter wizard

- Message Format: signal_event_format.se.dfdl.xsd
  The file we just created



New Signal Event Specification Wizard

Signal Event Specification Details

Specify event specification details. Default required fields: Event Name, Dispatch Adapters, Event Message Format DFDL File.

Event Name: my_event

Description:

Persistent: ● YES ○ NO ○ QUEUE_DEFAULT

Priority: -1

Expiry: -1

Enrichment Programs

Application Enrichment Program:

Dispatch Enrichment Program:

Dispatch Queue

● System Default Dispatch Queue

○ User Dispatch Queue

Error Processing

Error Queue:

Persistent: ● YES ○ NO ○ USE_EVENT_DEFINITION

Expiry: □ -1

Event Error Program:

Dispatch Adapters

my_DA

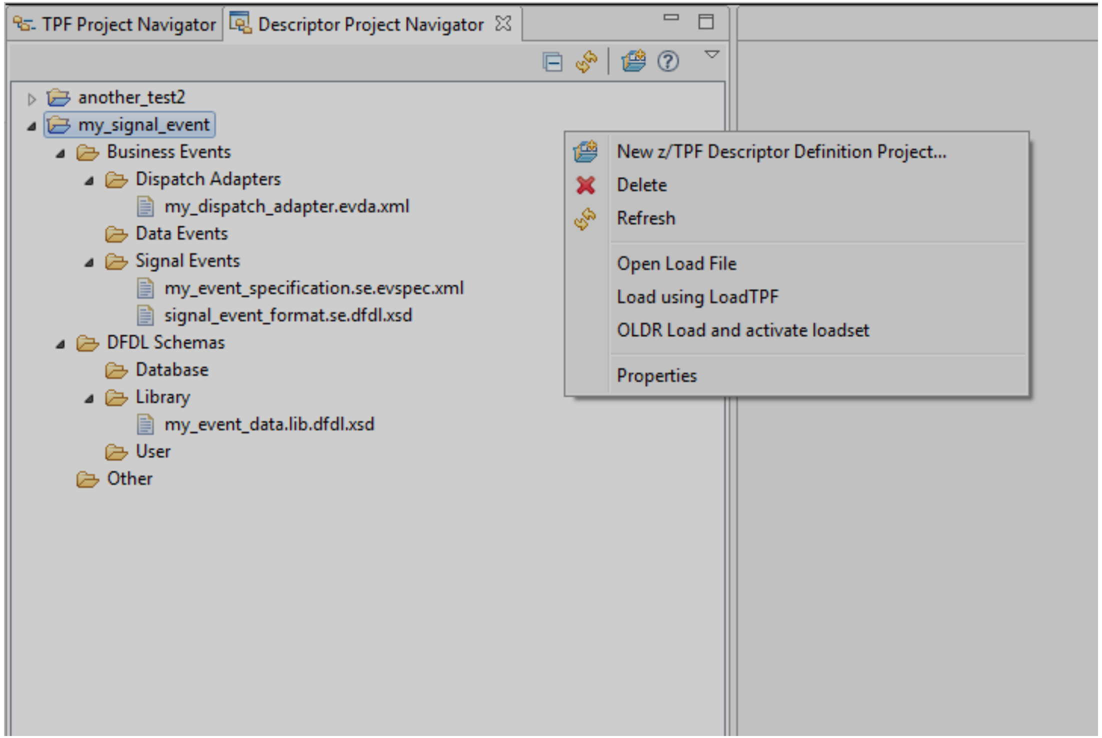
Message Format

Event Message Format DFDL File: signal_event_format.se.dfdl.xsd

DFDL editor

From the Descriptor Project Navigator able to:

- View load file for the project
- Load project using LoadTPF
- Use OLDR actions to load and activate project loadset


TPF Project Navigator
Descriptor Project Navigator
another_test2
my_signal_event
Business Events
Dispatch Adapters
my_dispatch_adapter.evda.xml
Data Events
Signal Events
my_event_specification.se.evspec.xml
signal_event_format.se.dfdl.xsd
DFDL Schemas
Database
Library
my_event_data.lib.dfdl.xsd
User
Other
New z/TPF Descriptor Definition Project...
Delete
Refresh
Open Load File
Load using LoadTPF
OLDR Load and activate loadset
Properties

Summary

- z/TPF Descriptor Definition Projects allow you to:
  - Organize and create z/TPF business event artifacts
  - Edit DFDL schemas with the DFDL editor
  - Test serialize and parse XML/binary data against a DFDL schema

Thank you!

Questions or comments?

Trademarks

- IBM, the IBM logo, ibm.com and Rational are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml.

Notes

- Performance is in Internal Throughput Rate (ITR) ratio based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput improvements equivalent to the performance ratios stated here.
- All customer examples cited or described in this presentation are presented as illustrations of the manner in which some customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics will vary depending on individual customer configurations and conditions.
- This publication was produced in the United States. IBM may not offer the products, services or features discussed in this document in other countries, and the information may be subject to change without notice. Consult your local IBM business contact for information on the product or services available in your area.
- All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.
- Information about non-IBM products is obtained from the manufacturers of those products or their published announcements. IBM has not tested those products and cannot confirm the performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.
- Prices subject to change without notice. Contact your IBM representative or Business Partner for the most current pricing in your geography.
- This presentation and the claims outlined in it were reviewed for compliance with US law. Adaptations of these claims for use in other geographies must be reviewed by the local country counsel for compliance with local laws.