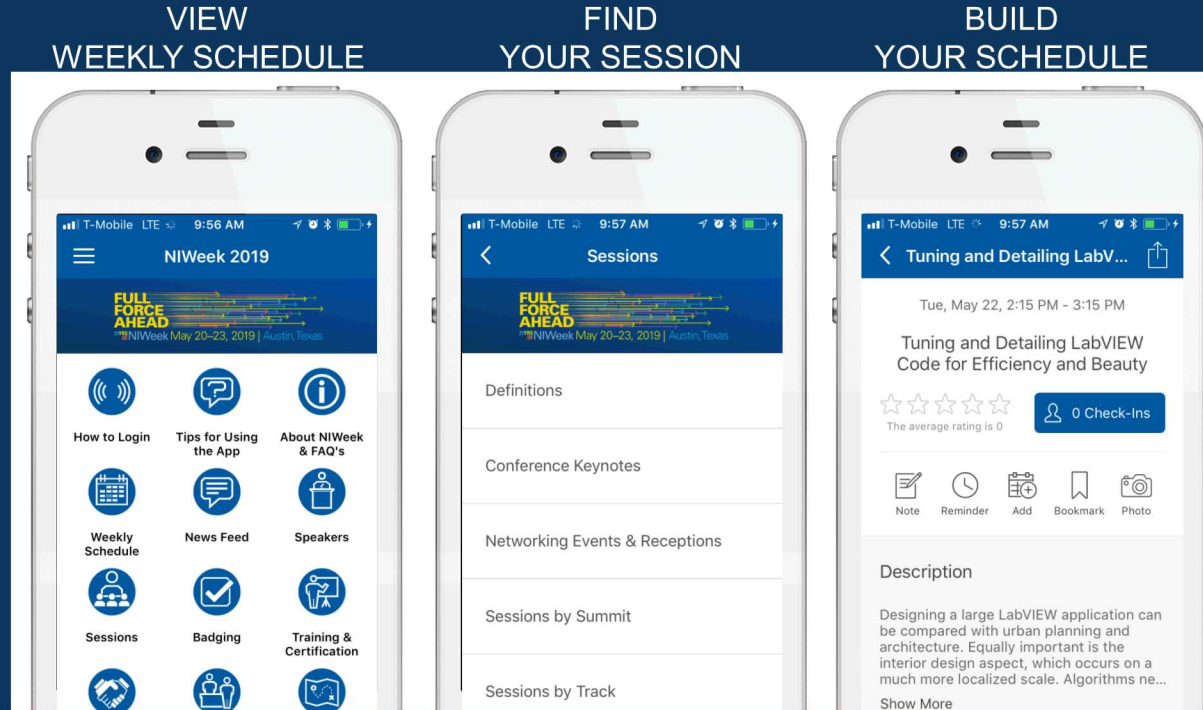


FULL FORCE AHEAD

25^{YRS}
OF NIWeek May 20–23, 2019 | Austin, Texas

Download and Login to the NIWeek Mobile App



More Information at ni.com/niweek



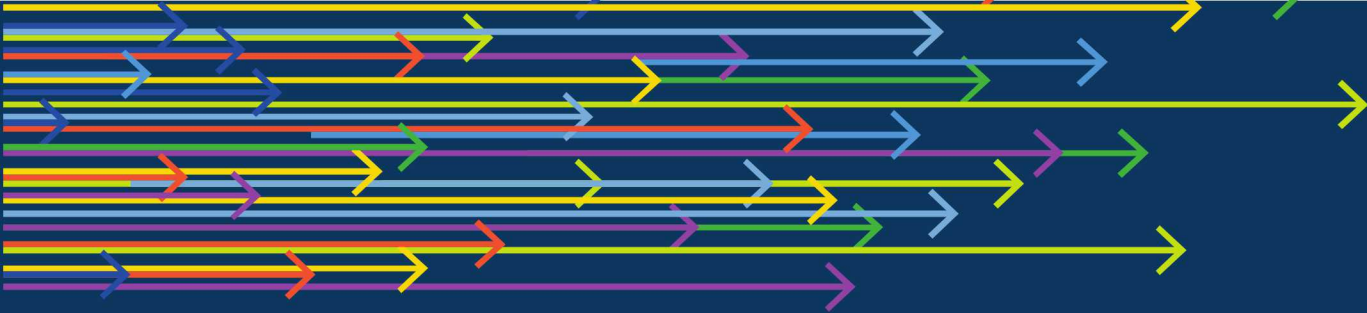


Scott Walkington

Sandia National Laboratory

Team Lead

CLD & CTD



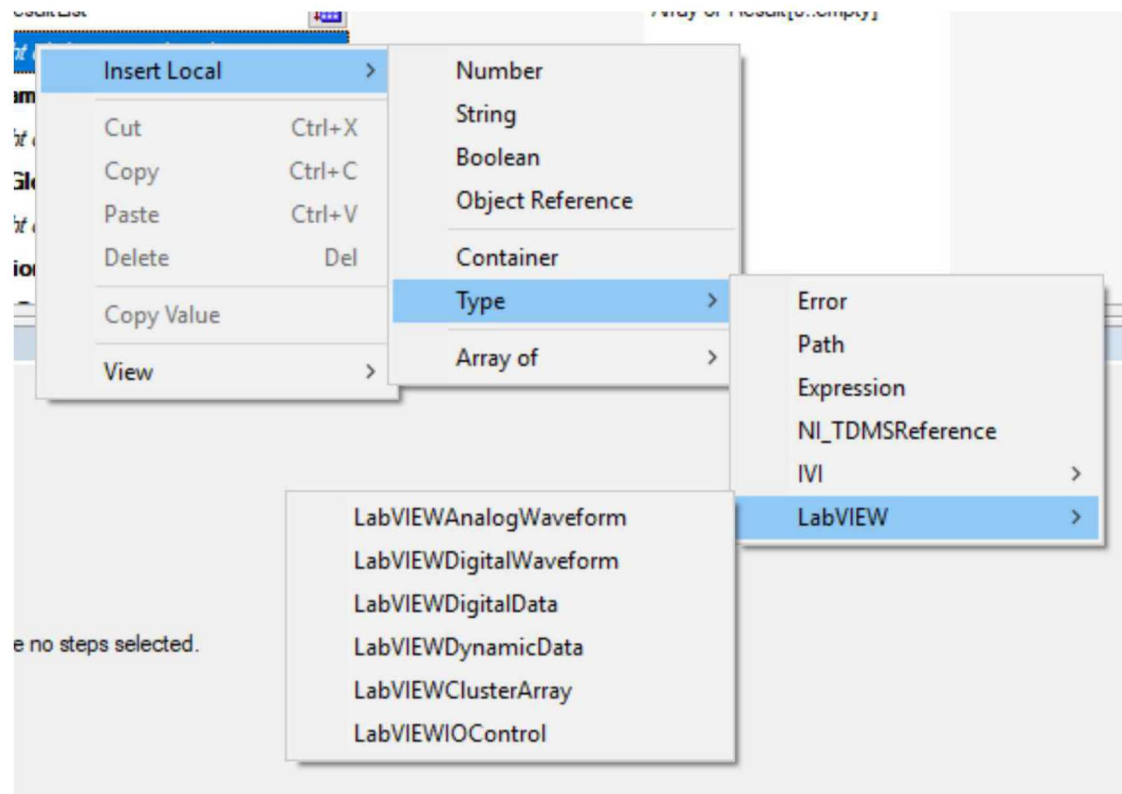
Novel Approaches to Pass Data Between TestStand and LabVIEW

Who has struggled with deciding what is the best approach to share/pass data between TestStand and LabVIEW? This talk will examine the pro's and con's of using variants, variants with attributes, objects, images, DVR's, clusters, pass by reference, pass by value, queues, and events to share/pass data between LabVIEW and TestStand.

Outline

- Pass by Value
- Pass by Reference
- Clusters
- Objects
- Variants
- Variants with Attributes
- Images
- DVR's
- Queues
- Events

Standard Variable Options



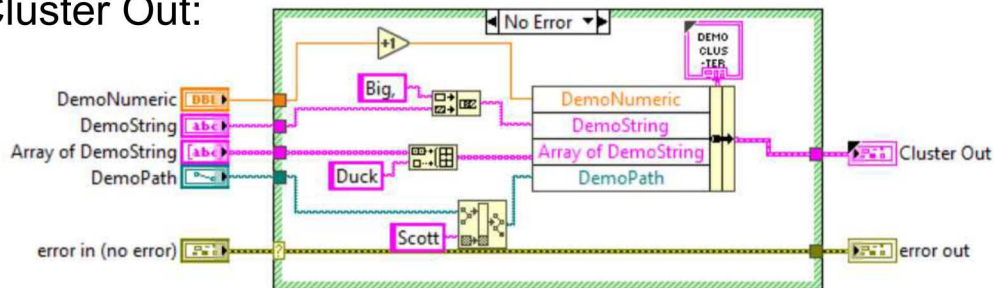
Basic Structure of LabVIEW VI's in Demo

The image displays the LabVIEW development environment with three main panels:

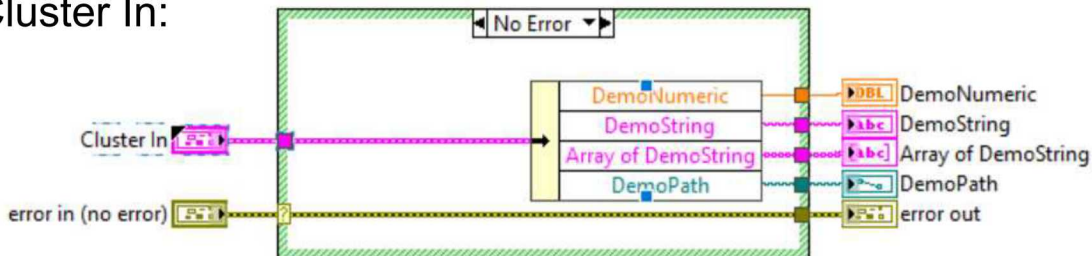
- Project Explorer (Top Left):** Shows the hierarchy of the VI. The 'Main' tab is active, containing a 'Cluster In' subVI. The 'Cluster In' subVI is highlighted, showing its internal structure: 'Message Popup', 'Cluster Out', 'Message Popup', and 'End Group'.
- Properties Window (Bottom Left):** Shows the 'Text and Buttons' tab. The 'Title Expression' is set to 'NameOf(Step)'. The 'Message Expression' is set to 'Number of ' + Parameters.DemoString + 's: ' + Str(Parameters.DemoNumeric, '%d')'. Below the expressions, a note states: 'Button Label Expressions (If unlabeled buttons are hidden)'.
- Locals and Parameters (Right):** This panel shows the 'Locals' and 'Parameters' clusters. The 'Locals' cluster contains 'ResultList' and 'Cluster'. The 'Parameters' cluster contains 'DemoPath', 'DemoNumeric', 'DemoString', and 'ArrayOfDemoString'. To the right of these clusters, a list of variables and their data types is shown: 'Array of Result[0..empty]', 'Cluster_Out (Container)', 'String (By reference)', 'Number (By reference)', 'String (By reference)', and 'Array of Strings[0..9] (By ...'.

Basic Structure of TestStand Sequence in Demo

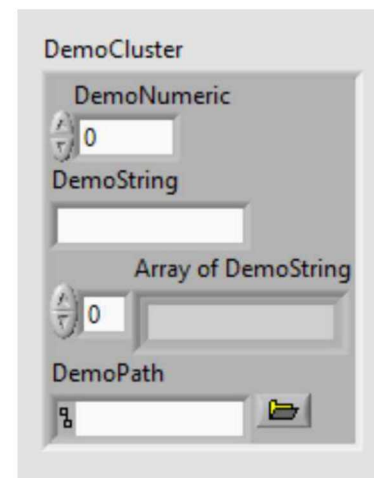
Cluster Out:



Cluster In:



Cluster:



Pass by Value

- Parameter value passed into subsequence does not get passed out

Message Popup	NameOf(Step)
PassByVal	Call PassByVal in NI_Week_Dem...
Message Popup	NameOf(Step)
PassByRef	Call PassByRef in NI_Week_De...
Message Popup	NameOf(Step)

- Pro or Con: Variable of Calling Sequence stays the same

Parameter Name	Type	Log	Default	Value		How Passed	C
DemoNumeric	Number	☐	☐	Parameters.DemoNumeric		☒ by value	
DemoString	String	☐	☐	Parameters.DemoString		☒ by value	

Subsequence:

Main (3)	
Change Dogs to Cats	Parameters.DemoString = "Cats"
Increment Number	Parameters.DemoNumeric = Parameters.DemoNumeric + 1
Message Popup	NameOf(Step)
<End Group>	

Pass by Reference

- Default setting in TestStand for all variables

Message Popup	NameOf(Step)
PassByVal	Call PassByVal in NI_Week_Dem...
Message Popup	NameOf(Step)
PassByRef	Call PassByRef in NI_Week_De...
Message Popup	NameOf(Step)

- Pro or Con: Variable of Calling Sequence is updated

Parameter Name	Type	Log	Default	Value		How Passed
DemoNumeric	Number	☐	☐	Parameters.DemoNumenc	 	by reference
DemoString	String	☐	☐	Parameters.DemoString	 	by reference

Subsequence:

Main (3)	
 Change Dogs to Cats	Parameters.DemoString = "Cat"
 Increment Number	Parameters.DemoNumeric = Parameters.DemoNumeric + 1
 Message Popup	NameOf(Step)
<End Group>	

Clusters

- Need to “Create/Update Custom Data Type from Cluster”
- Good for passing data in Sequential steps

+ error in (no error)	Container		in	☐
+ Cluster Out	Container		out	☐
+ error out	Container		out	☐

DemoCluster

DemoNumeric
0

DemoString
[Text Field]

Array of DemoString
0 [Text Field]

DemoPath
[File Picker]

Complex Cluster:

Create/Update Custom Data Type from Cluster

Use this dialog box to create or update a TestStand custom data type based on the LabVIEW cluster you select.

☒ Create a New Type ☐ Update an Existing Type

Type Name: DemoCluster

☐ Force Exact Match

LabVIEW Label	TestStand Name	LabVIEW Type	TestStand Type
DemoNumeric	DemoNumeric	Number (DBL)	Number
DemoString	DemoString	ASCII String	String
Array of DemoString	Array_of_DemoString	1D Array (ASCII)	Array of Strings
DemoPath	DemoPath	Path	String
DemoQueue out	DemoQueue_out	Number (U32)	Number
data value reference (data_value_reference)	Number (U32)	Number	Number
Object in TestStand (Object_in_TestStand)	Object Reference	Object Reference	Object Reference

Create Custom Data Type In File:
NI_Week_Demo.seq - <Selected Sequence File>

Help Create Cancel

Create/Update Custom Data Type from Cluster

Use this dialog box to create or update a TestStand custom data type based on the LabVIEW cluster you select.

☒ Create a New Type ☐ Update an Existing Type

Type Name: Cluster_Out

☐ Force Exact Match

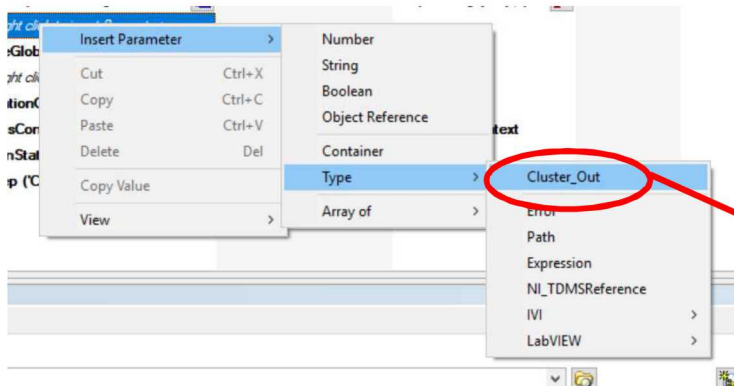
LabVIEW Label	TestStand Name	LabVIEW Type	TestStand Type
DemoNumeric	DemoNumeric	Number (DBL)	Number
DemoString	DemoString	ASCII String	String
Array of DemoString	Array_of_DemoString	1D Array (ASCII)	Array of Strings
DemoPath	DemoPath	Path	String

Create Custom Data Type In File:
NI_Week_Demo.seq - <Selected Sequence File>

Help Create Cancel

Clusters Continued

- Custom Data types appear in Insert -> Type-> Custom Data Type

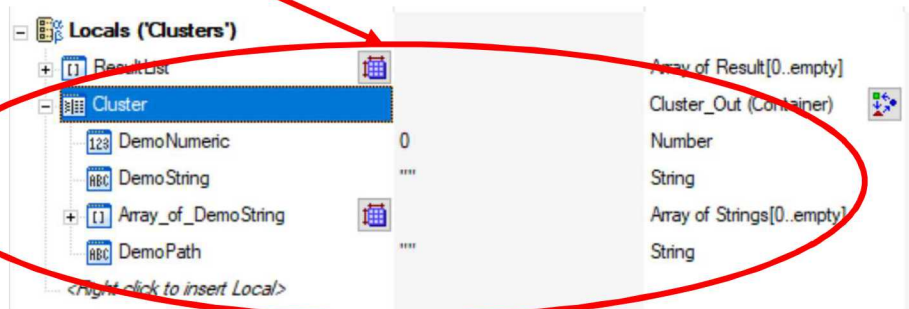


Pro's:

- Data easily visible in TestStand
- Custom data type can be used easily in multiple places
- Better to use Custom Data Type then creating a container with variables
 - Changes to a Custom Data Type will populate throughout sequence

Con's:

- If using just a container and you change the cluster then you must change the container everywhere
- Not very flexible if still making lots of changes in Cluster/Type def.



Objects

- Object Oriented Programming fully supported in TestStand
- Ok for passing data in Sequential steps

The screenshot shows the 'Step Settings for Object Out' dialog box. The 'Call Type' is set to 'Class Member Call'. The 'Class Path' is 'Object in TestStand.lvclass' and the 'Member Name' is 'Object Out.vi'. The 'Locals' panel shows 'ResultList' and 'Object' (circled in red). A red arrow points from the 'Object' local to the 'Object in TestStand.lvclass' parameter in the table below.

Parameter Name	Type	In/Out	Log	Default	Value
DemoPath	Path	in	☐	☐	Parameters.DemoPath
DemoNumeric	Number (DBL)	in	☐	☐	Parameters.DemoNumeric
DemoString	ASCII String	in	☐	☐	Parameters.DemoString
Array of DemoString [0.....	1D Array (AS...	in	☐	☐	Parameters.ArrayOfDemoString
error in (no error)	Container	in	☐	☒	
Object in TestStand.lvcl...	Object Reference	out	☐		Locals.Object
error out	Container	out	☐		Step.Result.Error

Locals ('Objects')
+ ResultList
+ Object
<Right click to insert Local>

Nothing
Array of Result[0..empty]
Object Reference

Pro's:

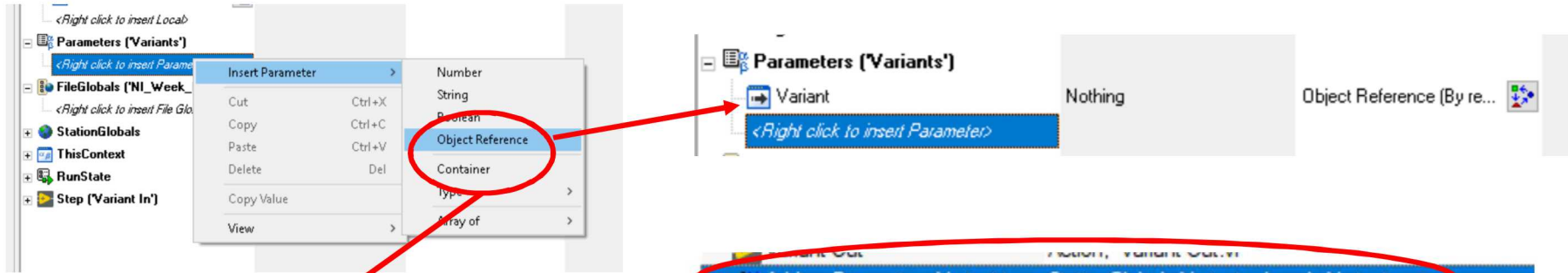
- Supports Dynamic dispatch
- Encapsulates Data
- Cannot accidentally change data in TestStand
- Good for passing data in Sequential steps

Con's:

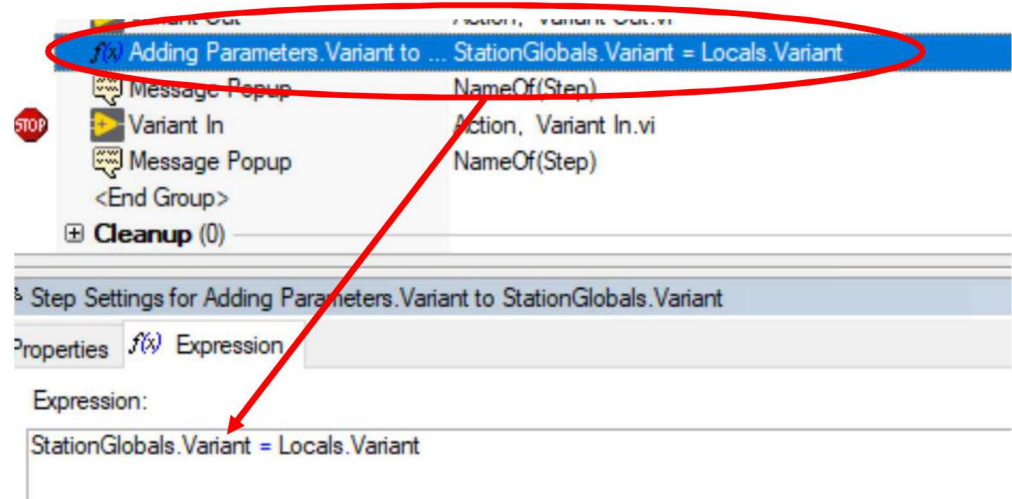
- Can not see what is contained in object during execution
- Can not change data in TestStand

Variants

- Initially, Create an Object Reference or Container

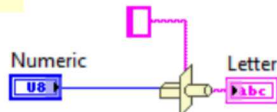
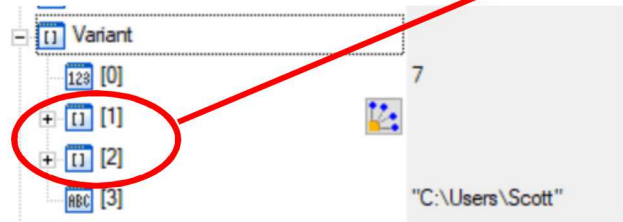
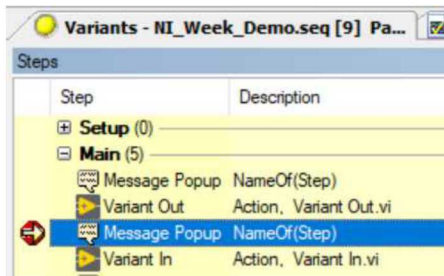


- For troubleshooting create variant in StationGlobals

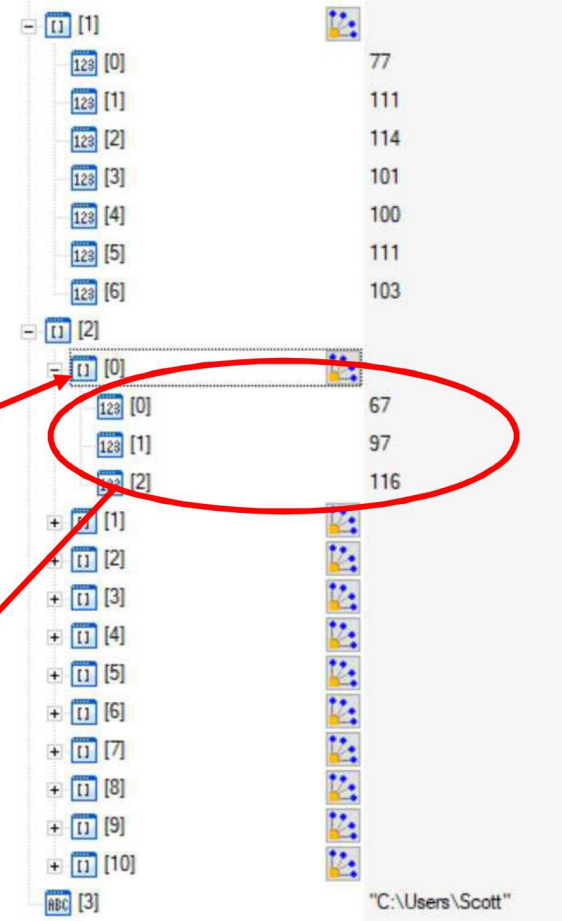


Variants Continued

- During runtime Container/Object Reference changes to “Array of Results”
- Variant Values can only be seen in variables tab during execution
 - Variant created in StationGlobals will retain values after run for trouble shooting



67 97 116
"D" "o" "g"



Variants

Pro's:

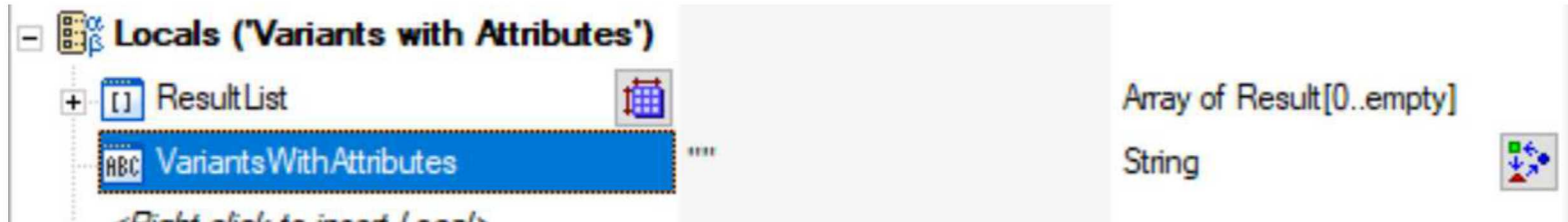
- Works well for sequential steps
- Provides some idea of data contained
- Very flexible if you are still changing/deciding the data to pass between steps

Con's:

- Can be easily overwritten
- Can be confusing when working with large clusters in the variant
- Can be confusing when passing around strings

Variants with Attributes

- Create String Variable

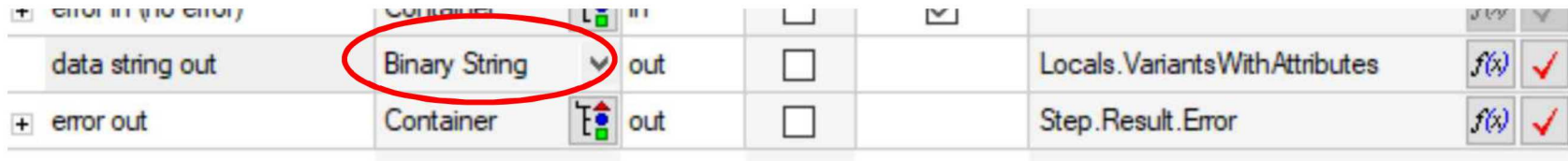


Locals (Variants with Attributes)

ResultList Array of Result[0..empty]

VariantsWithAttributes String

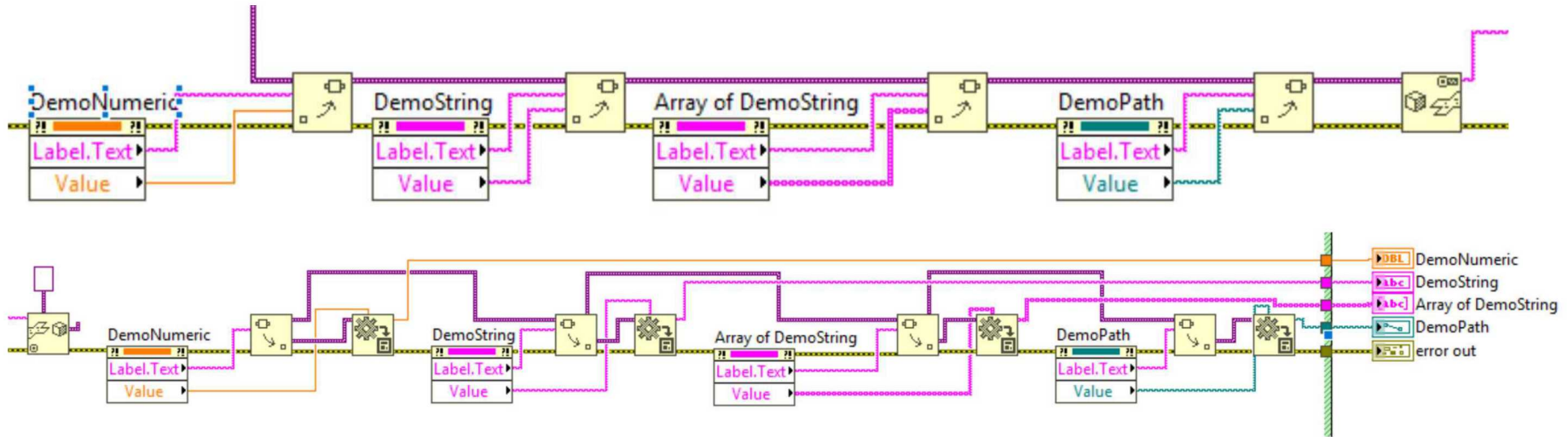
- MUST select Binary String



	Container	out				
data string out	Binary String	out	☐		Locals.VariantsWithAttributes	f(x) ✓
error out	Container	out	☐		Step.Result.Error	f(x) ✓

Variants with Attributes

- Must Use “Flatten to String”
 - Don’t use “Variant to Flatten String”
 - The help says this function strips out all variant attributes.
 - Guessing this is what TestStand is Leveraging under the hood



Variants with Attributes



Pro's:

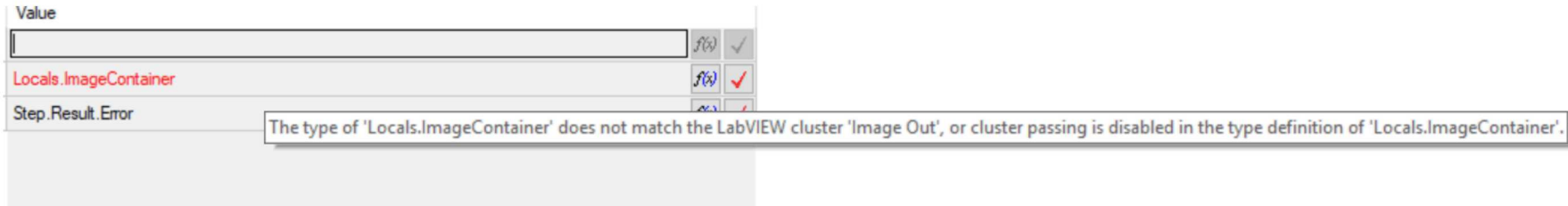
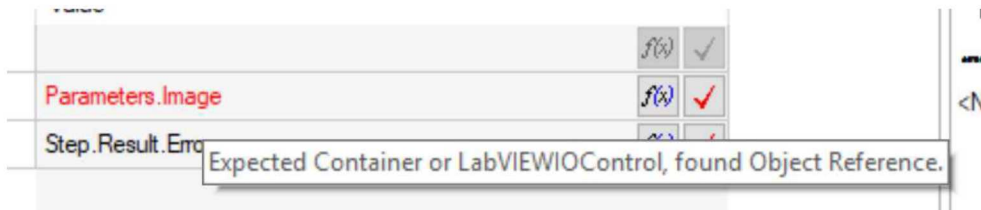
- Works ok for sequential steps
- Very flexible if you are still changing/deciding the data to pass between steps
- Handy for Variant Attribute Look up tables

Con's:

- Forget to change from ASCII string to Binary String
- Previous Variant stuff does not work.
 - Strips the attribute
- Long String of junk

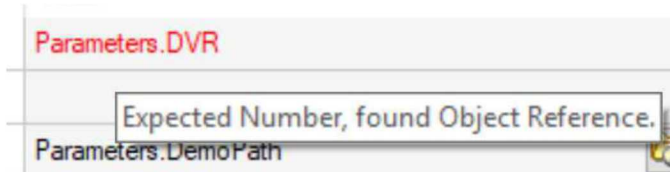
Images

- Need NI Vision
- Could only get the LabVIEWWIOControl to work
 - Not sure why the Container did not work



DVR's

- Pass DVR as a Number
- Pass Reference to DVR as a Number
- Pass DVR as a Data String like variant attribute

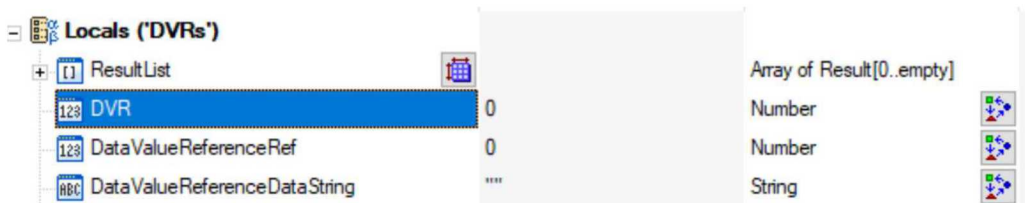


Pro's:

- Can be nice for large data sets

Con's:

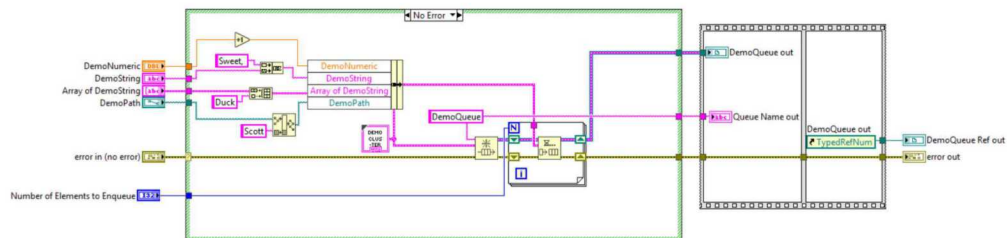
- Forget to change from ASCII string to Binary String
- Long String of junk



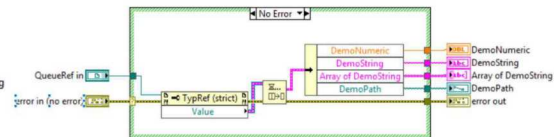
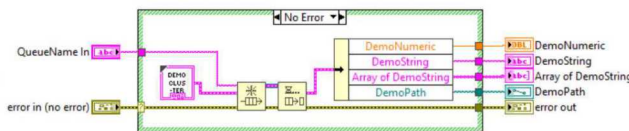
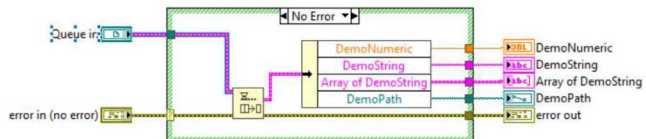
Queues

- Pass Queue as a Number
- Pass Reference to Queue as a Number
- Pass Queue as a Data String like variant attribute
- Pass Queue Name

Parameters.Queue		
Expected Number, found Object Reference.		
Parameters.DemoPath		



Locals ('Queues')	
ResultList	
123 Queue	0
ABC QueueName	error
123 QueueRef	0



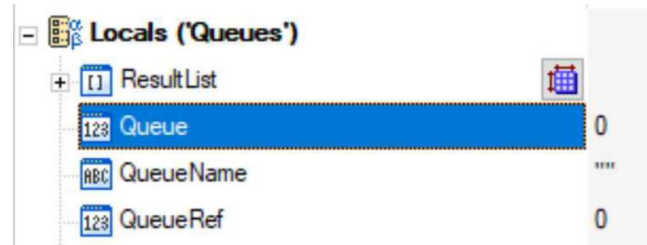
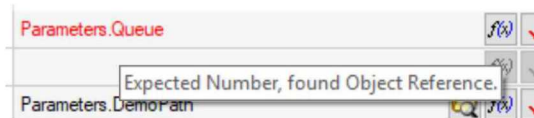
Queues Continued

Pro's:

- Can store multiple pieces of data
 - Can be from multiple steps
 - Can then be processed in a single step

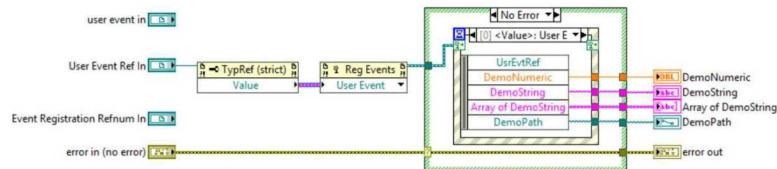
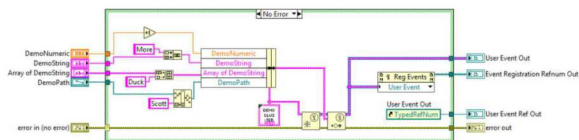
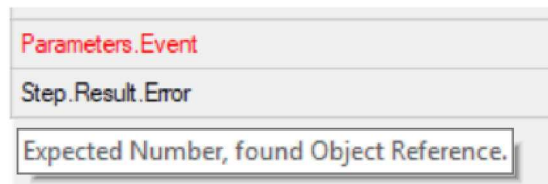
Con's

- Must make sure you enqueue enough elements otherwise code will hang waiting to dequeue element
- Can be confusing where elements are enqueue if you have many asynchronously running VI's



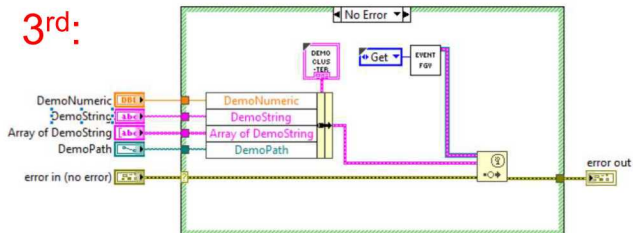
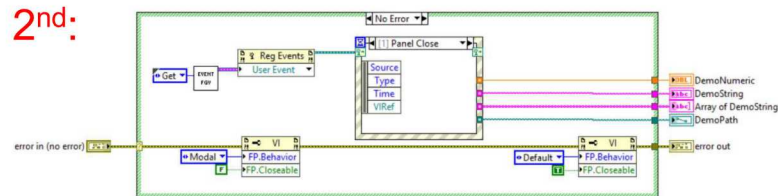
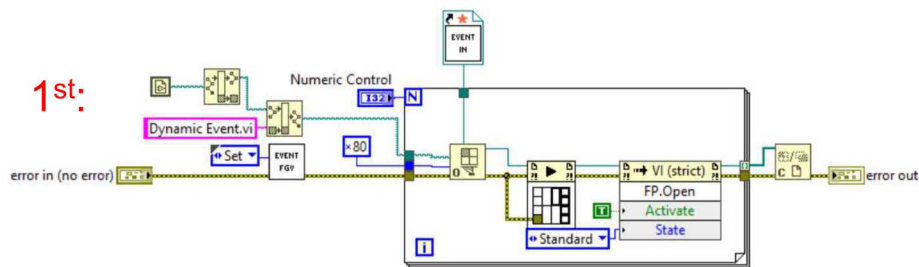
Events

- Cannot pass data between steps in traditional sense
 - Events are not like queues. Once the event fires if nothing is a registered listener then the data is gone.
- Must have a Asynchronously running VI registered to listen to event.
 - Can have as many clones running as you would like
 - All will simultaneously receive data.



Asynchronous Dynamic Events

- Can easily do One to Many



Conclusion

- Can put almost anything into an Object, Cluster, Variant or Data String and pass it between steps
- LabVIEW fully supports Object Orientated Programming
- Queues can be used, just make sure to enqueue enough elements
- Dynamic Events can easily send messages/data to many VI's simultaneously
- Variants work well for sequential steps or Dynamic dispatch
- Varying abilities to view data in TestStand
 1. High – Cluster
 2. Medium – Variant
 3. Low – Object, Data String, Queues, DVRs,

4G LTE 12:54 PM

Surveys

Title
Processing at the Edge: Why a Platform-Based Approach Is Ideal for the IIoT

Time
Tuesday, 1:00 PM - 2:00 PM

Speaker(s)
Nick Butler

Nick Butler

*1. Please rate the session content on the following

Overall Quality
- select one -

Technical Level
- select one -

Relevance to your job
- select one -

Relevance to published title and abstract
- select one -

Nick Butler

Navigation icons: Refresh, Back, Forward, Home, App Drawer

Before you go,
take the survey.

Slide Divider Topic Place Here

Stay Connected During and After NIWeek



ni.com/niweekcommunity



facebook.com/NationalInstruments



twitter.com/niglobal



youtube.com/nationalinstruments