IADS Configuration File Application Programming Interface

April 2008

SYMVIONICS Document SSD-IADS-012

Table of Contents

1. Introduction

The IADS Configuration data base is an ASCII file that is used by all the IADS software components for setup, communication, and archiving of both system and user generated meta-data. It is used and manipulated by both the IADS Compute Data Server (CDS) and the IADS Client display workstation (the Client).

1.1. Purpose

This document provides the interface specification necessary in order to manipulate the Configuration file using the IADS Configuration File Application Programmers Interface (API). This API was developed to allow outside agencies a programmatic interface in order to create and manipulate the IADS Configuration database file in a non real-time environment.

1.2. Scope

This document will cover the contents and structure of the IADS Configuration database file. In addition a general discussion of the API's Parameter Defaults table and the Event Marker table along with a discussion of the general SQL query interface will be included.

2. IADS Application Programming Interface

IADS provides a Component Object Model (COM) programming Application Programmers Interface (API) Dynamic Link Library (DLL) interface to manipulate an IADS Configuration file. This API provides methods using two different techniques; a Collection based interface for frequently used tables (see Section 3) and a general purpose SQL interface for complete access to all internal tables(See Appendix A for list of tables)

The COM DLL and test project are available for download on the SYMVIONICS, Inc. Website. Download item #10 under the "Programming Examples" section, here is the link to the Downloads page:

http://www.symvionics.com/products/IADS/downloads.html

3. Configuration Interface

The Configuration interface provides all the methods needed to manipulate the Configuration file. This is the API starting-point for all Collection interfaces (see Section 4) and access to the general purpose query engine (see Section 5). Following is a table of the methods available:

Method	*Return*	*Argument*	*Description*
Open	In	BSTR	Open an existing configuration file with create backup option
Create	In	BSTR	Create a new configuration file with open option
	In	VARIANT_BOOL	Set to true to Open the file after creation
Save	None	VOID	Commit and save changes from open configuration file
Close	In	VARIANT_BOOL	Close configuration file with save option
VersionFromFile	In	BSTR	Configuration file name and path
	Out	BSTR*	Version number
Version	Out	int*	return version of currently opened configuration file, (-1) indicates no version has been set yet
Version	In	int	put version of configuration file to create
OpenMessageLog	In	BSTR	Log error messages to a file
	In	VARIANT_BOOL	True to delete existing log
ParameterSets	Out	IParameterSets**	Get the ParameterSets collection
ParameterDefaults	Out	IParameterDefaults**	Get the ParameterDefaults collection
Events	Out	IEvents**	Get the Events collection
Thresholds	Out	IThresholds**	Get the Thresholds collection
Testpoints	Out	ITestpoints**	Get the Testpoints collection
Selections	Out	ISelections**	Get the Selections collection
PlannedTestPoints	Out	IPlannedTestPoints**	Get the Planned Test Points collection
Query	In	BSTR	Query interface. Keywords are: Select, Update, Delete, and Create. Returns an array of BSTR results if applicable
	Out	VARIANT*	Array of query results

4. Collection Interfaces

Seven collection interfaces are provided as a layer on top of the SQL engine due to their frequent use. They are the ParameterSets, ParameterDefaults, EventMarkers, Selections, Thresholds, Testpoints and PlannedTestpoints.

Appendix B has the value for IADS defined data types and enumerations. Following is a description of each of the collection interfaces

4.1. ParameterSets Collection

IADS allows multiple Parameter Sets to be defined and enabled for processing. The ParameterSets collection is used to return ParameterSet Item which in turn contains a ParameterDefaults collection (see Section 3.3). Following are the functions in the ParameterSets collection:

Method	*Return*	*Arguments*	*Description*
Count	Out	long*	Gets the number of IParameterSet Items within the collection.
Add	In	BSTR	Add a new IParameterSet Item to this collection. Add a ParameterSet by name
	In	BSTR	The group name of the IParameterSet Item to add to this collection.
	In	VARIANT_BOOL	The active flag of the IParameterSet Item to add to this
	Out	IParameterSet**	Return value reference to the newly added IParameterSet Item.
Remove	In	VARIANT	Remove IParameterSet object by index or name from this collection
RemoveAll	None	VOID	Remove all IParameterSet Items from this collection.
Item	In	VARIANT	Return an IParameterSet Item by name (string) or index number (0..Count). IndexOrName
	Out	IParameterSet**	Return value, reference to the specified IParameterSet Item. Returns a Set object by name (string) or index number (0..Count)
SaveTable	*None*	*VOID*	*Save changes to this Collection*

4.2. ParameterSet Item

The Parameter Set item contains the Set name, the Group name, and an IsActive boolean if the set is active in IADS. It also contains the ParameterDefaults collection for further processing of Parameter information.

Method	*Return*	*Arguments*	*Description*
SetName	Out	BSTR*	Get the Parameter Set name
SetName	In	BSTR	Set the Parameter Set Name
Group	Out	BSTR*	Get Parameter Group name
Group	In	BSTR	Set Parameter Group name
IsActive	Out	VARIANT_BOOL *	Get Parameter Set Active
IsActive	In	VARIANT_BOOL	Set Parameter Set is active
ParameterDefaults	Out	IParameterDefaults *	Get Parameter Defaults collection

4.3. ParameterDefaults Collection

The ParameterDefaults Collection holds all the available Parameter Items. It returns ParameterDefault Item which contains all the available properties for each parameter.

Method	*Return*	*Arguments*	*Description*
Count	Out	long*	Gets the number of IParameterDefault items within the collection.
Add	In	BSTR	Add a new IParameterDefault Item to this collection, this is the Parameter Name
	In	IadsDataType	This is the inherent type of the parameter
	In	Double	This is the sample rate of the parameter
	Out	IParameterDefault**	Return value, reference to the newly added IParameterDefault Item.
Remove	In	VARIANT	IParameterDefault Item by index or name from this collection
RemoveAll	None	VOID	Remove all IParameterDefault items from this collection.
Item	In	VARIANT	Return an IParameterdefault Item by name (string) or index number
SetName	In	BSTR*	Return the IParameterSet name that this parameter belongs to
SaveTable	*None*	*VOID*	*Save changes to this collection*

4.4. ParameterDefault Item

The Parameter item is returned from the ParameterDefaults collection. Following are the Methods in the ParameterDefaults collection along with their return value, or input argument and description

Method	*Return*	*Argument Type*	*Description*
Name	Out	BSTR*	Get parameter name
Name	In	BSTR	Set parameter name
DataType	Out	IadsDataType*	Get parameter data type
DataType	In	IadsDataType	Set parameter data type
Group	Out	BSTR*	Get parameter group name
Group	In	BSTR	Set parameter group name
SubGroup	out	BSTR*	Get parameter subgroup name
SubGroup	In	BSTR	Set parameter subgroup name
ShortName	Out	BSTR*	Get parameter short name
ShortName	In	BSTR	Set parameter short name
LongName	Out	SBSTR*	Get parameter long name
LongName	In	BSTR	Set parameter long name
Units	Out	BSTR*	Get parameter units name
Units	In	BSTR	Set parameter units name
Color	Out	OLE_COLOR*	Get color value
Color	In	OLE_COLOR	Set color value
Width	Out	int*	Get pen width
Width	In	int	Set pen width
DataSourceType	Out	IadsDataSourceType*	Get the data source type
DataSourceType	In	IadsDataSourceType	Set data source type
DataSourceArgument	Out	BSTR*	Get the data source argument
DataSourceArgument	In	BSTR	Set data source argument
SampleRate	out	double*	Get the data sample

			rate
SampleRate	In	double	Set data sample rate
LoadLimitNegative	Out	double*	Get negative design load limit
LoadLimitNegative	In	double	Set negative design load limit
LoadLimitPositive	Out	double*	Get positive design load limit
LoadLimitPositive	In	double	Set positive design load limit
TimeScaleRangeMin	Out	double*	Get time scale range min
TimeScaleRangeMin	In	double	Set time scale range min
TimeScaleRangeMax	Out	double*	Get time scale range max
TimeScaleRangeMax	In	double	Set time scale range max
TimeCaleAuto	in	double	Not currently used in IADS
TimeScaleAuto	Out	IadsOnOrOff*	Get time scale auto (Not currently used by IADS)
TimeScaleAuto	In	IadsOnOrOff	Set time scale auto (not currently used by IADS)
FreqScaleRangeMin	Out	double*	Get frequency scale range Max
FreqScaleRangeMin	In	double	Set frequency scale range Min
FreqScaleRangeMax	Out	double*	Get frequency scale range max
FreqScaleRangeMax	In	double	Set frequency scale range max
FreqScaleAuto	In	double	Not currently used in IADS
FreqScaleAuto	Out	IadsOnOrOff*	Get frequency scale auto (not curently used by IADS)
FreqScaleAuto	In	IadsOnOrOff	Set frequency scale auto (not curently used by IADS)
WarningThreshRangeMin	Out	double*	Get warning threshold min range
WarningThreshRangeMin	In	double	Set warning threshold range min

WarningThreshRangeMax	Out	double*	Get warning thresh range max
WarningThreshRangeMax	In	double	Set Get warning thresh range max
WarningThreshColor	Out	OLE_COLOR*	Get Warning Threshold color
WarningThreshColor	In	OLE_COLOR	Set warning threshold color
WarningThreshLabel	Out	BSTR*	Get warning threshold label
WarningThreshLabel	In	BSTR	Set warning threshold label
WarningThreshLineWidth	Out	int*	Get warning threshold line width
WarningThreshLineWidth	In	int	Set warning threshold line width
AlarmThreshRangeMin	Out	double*	Get alarm threshold range min
AlarmThreshRangeMin	In	double	Set alarm threshold range min
AlarmThreshRangeMax	Out	double*	Get alarm threshold range max
AlarmThreshRangeMax	In	double	Set alarm threshold range max
AlarmThreshColor	Out	OLE_COLOR*	Get alarm threshold color
AlarmThreshColor	In	OLE_COLOR	Set alarm threshold color
AlarmThreshLabel Out	Out	BSTR*	Get alarm threshold label
AlarmThreshLabel	In	BSTR	Set alarm threshold label
AlarmThreshLineWidth	Out	int*	Get alarm threshold line width
AlarmThreshLineWidth	In	int	Set alarm threshold line width
FilterActive	Out	IadsYesOrNo*	Get filter active
FilterActive	In	IadsYesOrNo	Set filter active
FilterAlgorithm	Out	IadsFilterAlgorithm*	Get filter algorithm
FilterAlgorithm	In	IadsFilterAlgorithm	Set filter algorithm
FilterPassType	Out	IadsFilterPassType*	Get filter pass type
FilterPassType	In	IadsFilterPassType	Set filter pass type
FilterLowCutoff	Out	double*	Get filter low cutoff
FilterLowCutoff	In	double	Set filter low cutoff

FilterHighCutoff	Out	double*	Get filter high cutoff
FilterHighCutoff	In	double	Set filter high cutoff
FilterOrder	Out	int*	Get filter order
FilterOrder	In	int	Set filter order (1..8)
WildPointRangeMin	Out	double*	Get wild point range min
WildPointRangeMin	In	double	Set wild point range min
WildPointRangeMax	Out	double*	Get wild point range max
WildPointRangeMax	In	double	Set wild point range max
WildPointCorrectionMethod	Out	IadsDataCorrectionMethod*	Get wild point correction method
WildPointCorrectionMethod	In	IadsDataCorrectionMethod	Set wild point correction method
WildPointCorrectionValue	Out	double*	Get wild point correction value
WildPointCorrectionValue	In	double	Set wild point correction value
SignChange	Out	IadsYesOrNo*	Get sign change
SignChange	In	IadsYesOrNo	Set sign change
NullCorrection	Out	IadsNullCorrection*	Get null correction type
NullCorrection	In	IadsNullCorrection	Set null correction, depends on data source type if TPP(nullOn or nullOff), if derived(equationResult, equationInput
NullBaseline	Out	double*	Get null basline value
NullBaseline	In	double	Set null baseline value
NullBias	Out	double*	Get null base value that is added to the parameter value
NullBias	In	double	Set null baseline value that is added to the parameter value
NullGroup	Out	IadsNullGroup*	Get Parameter Null

			group
NullGroup	In	IadsNullGroup	Set Parameter Null group
SpikeDetectionMethod	Out	IadsSpikeDetectionMethod*	Get spike detection method
SpikeDetectionMethod	In	IadsSpikeDetectionMethod	Set spike detection method
SpikeCorrectionMethod	Out	IadsDataCorrectionMethod*	Get spike correction method (last good value or none
SpikeCorrectionMethod	In	IadsDataCorrectionMethod	Set spike correction Method (last good value or none)
SpikeChangeLimit	Out	double*	Get spike change limit
SpikeChangeLimit	In	double	Set spike change limit
ComputeType	Out	IadsComputeType*	Get default compute type
ComputeType	In	IadsComputeType	Set default compute type
ExcitationSignal	Out	BSTR*	Get default excitation parameter
ExcitationSignal	In	BSTR	Set default excitation parameter
WindowType	Out	IadsWindowType*	Get window type
WindowType	In	IadsWindowType	Set window type
Alpha	Out	IadsAlpha*	Get alpha value for kaiser-bessel Window
Alpha	In	IadsAlpha	Set alpha value for kaiser-bessel window type
AveragingMethod	Out	IadsAverageMethod*	Get avergaing method, (avgTime not implemented
AveragingMethod	In	IadsAverageMethod	Set avergaing method, (avgTime not implemented)
Overlap	Out	double*	Get overlap value (0.0 >= value < 100.0
Overlap	In	double	Set overlap value

			(0.0 >= value < 100.0
BlocksPerAverage	Out	int*	Get blocks per average (1..5)
BlocksPerAverage	In	int	Set blocks per average (1..5)
BlockSize	Out	IadsBlockSize*	Get block size - (64 bytes..64K bytes
BlockSize	In	IadsBlockSize	Set block size - (64 bytes.64K bytes
StringLookupTable	Out	BSTR*	Get string lookup table
StringLookupTable	In	BSTR newVal	Set String Lookup table
Delete	Out	VARIANT_BOOL*	Get Parameter deletion setting
Delete	In	VARIANT_BOOL	Set parameter deletion setting

4.5. Events Collection

Method	*Return*	*Arguments*	*Description*
Count	Out	long*	Gets the number of IEvent Items within the collection.
Add	In	BSTR	Add a new IEvent Item to this collection by parameter name.
	In	BSTR	IRIG time at threshold break.
	Out	IEvent**	Return value reference to the newly added IEvent Item.
Remove	In	VARIANT	Remove IEvent Item by index or name from this collection
RemoveAll	None	VOID	Remove all IEvent Items from this collection.
Item	In	VARIANT	Return an IEvent Item by name (string) or index number (0..Count). IndexOrName
	Out	IThreshold**	Return value, reference to the specified IEvent Item. Returns a Set object by name (string) or index number (0..Count)
SaveTable	*None*	*VOID*	*Save changes to this Collection*

4.6. Event Item

Method	*Return*	*Argument*	*Description*
Group	Out	BSTR*	Get the group name
Group	In	BSTR	Set the group name
SubGroup	Out	BSTR*	Get subgroup name
SubGroup	In	BSTR	Set subgroup name
User	Out	BSTR*	Get user name
User	In	BSTR	Set user name
IrigTime	Out	BSTR*	Get IRIG time
IrigTime	In	BSTR	Set IRIG time
Name	Out	BSTR*	Get the comment
Name	In	BSTR	Set a comment
PropBag	Out	BSTR*	Get property bag
PropBag	In	BSTR	Set the property bag

4.7. Thresholds Collection

Method	*Return*	*Arguments*	*Description*
Count	Out	long*	Gets the number of IThreshold Items within the collection.
Add	In	BSTR	Add a new IThreshold Item to this collection by parameter name.
	In	BSTR	IRIG time at threshold break.
	In	IadsThresholdLevel	Threshold Level at the break
	Out	IThreshold**	Return value reference to the newly added

			IThreshold Item.
Remove	In	VARIANT	Remove IThreshold Item by index or name from this collection
RemoveAll	None	VOID	Remove all IThreshold Items from this collection.
Item	In	VARIANT	Return an IThreshold Item by name (string) or index number (0..Count). IndexOrName
	Out	IThreshold**	Return value, reference to the specified IThreshold Item. Returns a Set object by name (string) or index number (0..Count)
SaveTable	*None*	*VOID*	*Save changes to this Collection*

4.8. Threshold Item

Method	***Return***	***Argument***	***Description***
Group	Out	BSTR*	Get the group name
Group	In	BSTR	Set the group name
SubGroup	Out	BSTR*	Get subgroup name
SubGroup	In	BSTR	Set subgroup name
User	Out	BSTR*	Get user name
User	In	BSTR	Set user name
Level	Out	IadsThresholdLevel*	Get Threshold Level
Level	In	IadsThresholdLevel	Set Thershold Level
AnalysisWindowName	Out	BSTR*	Get property bag
AnalysisWindowName	In	BSTR	Set the property bag
DisplayType	Out	BSTR*	Get property bag
DisplayType	In	BSTR	Set the property bag
ParameterName	Out	BSTR*	Get property bag
ParameterName	In	BSTR	Set the property bag
IrigTimeAtBreak	Out	BSTR*	Get IRIG time at threshold break
IrigTimeAtBreak	In	BSTR	Set IRIG time at threshold break
ValueAtBreak	Out	double*	Get value at time break
ValueAtBreak	In	double	Set IRIG time
DisplayName	Out	BSTR*	Get display name
DisplayName	In	BSTR	Set the display name
Comment	Out	BSTR*	Get the comment
Comment	In	BSTR	Set a comment
PropBag	Out	BSTR*	Get property bag
PropBag	In	BSTR	Set the property bag

4.9. Selections Collection

Method	*Return*	*Arguments*	*Description*
Count	Out	long*	Gets the number of ISelection Items within the collection.
Add	In	BSTR	Add a new ISelection Item to this collection by parameter name
	In	BSTR	IRIG time of selection.
	In	Double	Value of selection
	Out	ISelection**	Return value reference to the newly added ISelection Item.
Remove	In	VARIANT	Remove ISelection Item by index or name from this collection
RemoveAll	None	VOID	Remove all ISelection Items from this collection.
Item	In	VARIANT	Return an ISelection Item by name (string) or index number (0..Count). IndexOrName
	Out	ISelection**	Return value, reference to the specified ISelection Item. Returns the Item by name (string) or index number (0..Count)
SaveTable	None	VOID	Save changes to this collection

4.10. Selection Item

Method	*Return*	*Argument*	*Description*
Group	Out	BSTR*	Get the group name
Group	In	BSTR	Set the group name
SubGroup	Out	BSTR*	Get subgroup name
SubGroup	In	BSTR	Set subgroup name
User	Out	BSTR*	Get user name
User	In	BSTR	Set user name
IrigTime	Out	BSTR*	Get IRIG Time
IrigTime	In	BSTR	Set IRIG Time
Value	Out	double*	Get Selection Value
Value	In	double	Set Selection Value
Parameter	Out	BSTR*	Get Parameter name
Parameter	In	BSTR	Set Parameter Name
Filter	Out	BSTR*	Get Filter
Filter	In	BSTR	Set Filter
Display	Out	BSTR*	Get Display Name
Display	In	BSTR	Set Display Name
Comment	Out	BSTR*	Get Comment
Comment	In	BSTR	Set Comment
PropBag	Out	BSTR*	Get property bag
PropBag	In	BSTRS	Set the property bag

4.11. Testpoints Collection

Method	*Return*	*Arguments*	*Description*
Count	Out	long*	Gets the number of ITestpoint Items within the collection.
Add	In	BSTR	Add a new ITestpoint Item to this collection by test point string.
	In	BSTR	The start time of the test point
	In	BSTR	The stop time of the test point
	Out	ITestpoint**	Return value reference to the newly added ITestpoint Item.
Remove	In	VARIANT	Remove ITestpoint Item by index or name from this collection
RemoveAll	None	VOID	Remove all ITestpoint Items from this collection.
Item	In	VARIANT	Return an IThreshold Item by name (string) or index number (0..Count). IndexOrName
	Out	IThreshold**	Return value, reference to the specified IThreshold Item. Returns the Item by name (string) or index number (0..Count)
SaveTable	*None*	*VOID*	*Save changes to this Collection*

4.12. Testpoint Item

Method	*Return*	*Argument*	*Description*
Group	Out	BSTR*	Get the group name
Group	In	BSTR	Set the group name
SubGroup	Out	BSTR*	Get subgroup name
SubGroup	In	BSTR	Set subgroup name
User	Out	BSTR*	Get user name
User	In	BSTR	Set user name
Testpoint	Out	BSTR*	Get testpoint
Testpoint	In	BSTR	Set user name
Description	Out	BSTR*	Get testpoint
Description	In	BSTR	Set user name
Maneuver	Out	BSTR*	Get testpoint
Maneuver	In	BSTR	Set user name
StartTime	Out	BSTR*	Get Start time
StartTime	In	BSTR	Set Start Time
StopTime	Out	BSTR*	Get Stop time
StopTime	In	BSTR	Set Stop Time
PropBag	Out	BSTR*	Get property bag
PropBag	In	BSTR	Set the property bag

4.13. PlannedTestpoints Collection

Method	*Return*	*Arguments*	*Description*
Count	Out	long*	Gets the number of IPlannedtestpoints Items within the collection.
Add	In	BSTR	Add a new IPlannedTestpoint Item to this collection by unique testpoint string
	Out	IPlannedtestpoint**	Return value reference to the newly added IPlannedTestpoint Item.
Remove	In	VARIANT	Remove IParameterSet object by index or name from this collection
RemoveAll	None	VOID	Remove all IParameterSet Items from this collection.
Item	In	VARIANT	Return an IParameterSet Item by name (string) or index number (0..Count). IndexOrName
	Out	IParameterSet**	Return value, reference to the specified IParameterSet Item. Returns a Set object by name (string) or index number (0..Count)
SaveTable	None	VOID	*S*ave changes to this Collection

4.14. PlannedTestpoint Item

Method	*Return*	*Argument*	*Description*
Group	Out	BSTR*	Get the group name
Group	In	BSTR	Set the group name
SubGroup	Out	BSTR*	Get subgroup name
SubGroup	In	BSTR	Set Subgroup name
User	Out	BSTR*	Get User name
User	In	BSTR	Set User name
Testpoint	Out	BSTR*	Get Testpoint
Testpoint	In	BSTR	Set Testpoint (This is a user defined format)
Description	Out	BSTR*	Get Description
Description	In	BSTR	Set Description
Maneuver	Out	BSTR*	Get Maneuver name
Maneuver	In	BSTR	Set the Maneuver name
AircraftConfig	Out	BSTR*	Get the Aircraft Configuration (This is a user defined setting)
AircraftConfig	In	BSTR	Set the Aircraft Config
FlightConditions	Out	BSTR*	Get Flight Conditions (This is a user defined setting)
FlightConditions	In	BSTR	Set the Flight Conditions
PredictedResults	Out	BSTR*	Get the Predicted Results
PredictedResults	In	BSTR	Set the Predicted Results
PropBag	Out	BSTR*	Get Property bag
PropBag	In	BSTR	Set the Property bag

5. General Purpose Query Interface

From the IadsConfig API level general purpose queries can be made. This allows full access to the Configuration file without using the Collection interfaces as detailed above. Intimate knowledge of the File structure is required. Use caution with this routine, especially with hierarchical tables, as entries made will most likely need to be made in other relational tables as well

5.1. Query string construction has the following form:

"keyword <field name> from <table name> where <qualifiers>"

Keywords are:

1. Select - Selected values are returned in BSTR array

2. Update - Query user passes in a string to update in the configuration file.

3. Delete - Deletes one or more rows in a configuration file

4. Create - Create a table in the configuration file.

Field names: These are directly from the Configuration file; therefore the query user must have knowledge of its table construction: Multiple field names are separated by commas. A wild card of '*' can be used in which case the entire row is returned with individual field values delimited by the '|' vertical pipe character.

Table name: Table names are those that are in the configuration file, therefore the query user must have knowledge of its construction.

Qualifiers: Qualifiers take the form of: "field name = value". All values of type string must be enclosed in single quotes, example: Group = 'Loads'.

5.2. Query Examples

1) ***Select * from Desktops*** - Get all fields for all entries in a particular table.

2) ***Select * from Desktops where Group = 'Flutter'*** - Get all fields for selected entries using a "where" clause.

3) ***Select * from Desktops where Group = 'Flutter' && AnalysisWindowName = 'DoubleIntTest'*** - Get all fields for selected entries using a compound "where" clause.

4) ***Select SubGroup from Desktops where Group = 'Flutter' && AnalysisWindowName = 'DoubleIntTest'*** - Get one field for selected entries using a compound "where" clause.

5) ***Select System.RowNumber from Desktops*** - Get one field for selected entries using a compound "where" clause. Note usage of "System.RowNumber. This selects a row based on it's unique Id (be careful with this).

6) ***Update BogusDesktops set * = a|b|c|d|e*** - Modify all fields in all rows in a table

7) ***Update BogusDesktops set * = a|b|c|d|e where Group = 'Flutter'*** - Modify all fields in the specified row(s) based on a match in a single field.

8) ***Update BogusDesktops set SubGroup = 'Pat' where Group = 'FQ'*** - Modify a single field in a specified row.

9) ***Create table BogusTable (Group String, X Int, Y Int, AutoScale list(True, False), Classification list(high,medium,low))*** - Creates a table called BogusTable with fields as shown above. Note usage of the list type. This will create a drop down list in the configuration tool for easier user entry.

10) ***Delete * from BogusDesktops where Group = 'Flutter'*** - Delete every row from the BogusDesktopns table where the field value is Flutter

11) ***Delete * from BogusDesktops where System.RowNumber = 3*** - Delete using "built-in" unique system id or row number

6. Appendix A - IADS Configuration Table Reference

Name	Description	Type
AircraftReferences	Used by Parameter Identification	Flat
Aircraft Properties	Aircraft properties table	Flat
ActualFlutterTestPointsLog	Completed flutter test points	Flat
ActualLoadsTestPointsLog	Completed Loads test points	Flat
AnalysisLog	Location of saved analysis results	Flat
AnalysisWindows	User created Analysis Windows	Hierarchical
AttachedDataDisplays	List of displays attached to AWs	Hierarchical
FlutterSummaryLog	On going collection of flutter results	Flat
DataStorageInformation	Data Information from a real-time test	Flat
CurrentFlightInformation	Table of the current flight test	Flat
Classifications	List of available classifications	Flat
GroupDefinitions	List of available classification strings	Flat
PredefinedComments	Pre-defined event marker strings	Flat
Constants	User defined constants	Flat
DataDisplays	List of used data displays	Hierarchical
DataStorageLog	Information on data archive set	Flat
DataDropOutLog	Not Currently used	Flat
DisplayDefaults	Data display defaults	Flat
Desktops	User defined Desktops	Hierarchical
ExtendedDesktopInfo	Additional Desktop information	Flat
Envelopes	User defined envelopes	Flat
ReferenceCurves	User defined reference envelopes	Flat
EventMarkerLog	User created event markers	Flat
HardCopyLog	<blank>	Flat
HardCopyBanners	<blank>	Flat
LogBehavior	Log behavior settings	Flat
LoadsSummaryLog	User created loads information	Flat
TableUpdateBehavior	Table update behavior properties	Flat
ModalDefinitions	User defined mode ranges and titles	Flat
ParameterDefaultsState	List of parameter default sets	Hierarchical
ParameterDefaults	List of all user defined parameters	Flat
ParametersSavedInDisplays	Parameters saved in defined displays	Hierarchical
PlannedLoadsTestPoints	User defined planned loads test points	Flat
PlannedFlutterTestPoints	User defined Planned Flutter points	Flat
AlphaNumeric	List of Alphanumeric displays	Flat
AlphaNumericTable	List of Alphanumerictable displays	Flat
Annunciator	List of Annunciator displays	Flat
FrequencyResponsePlot	List of Frequency response displays	Flat
DisplayLabel	List of Display Label displays	Flat
DisplayFolder	List of Display Folder displays	Flat
CrossPlot	List of Cross Plot displays	Flat
DisplayTab	List of Display Tab displays	Flat

FlutterSummaryPlot	List of Flutter Summary Plot displays	Flat
FrequencyPlot	List of Frequency Plot displays	Flat
LoadsSummaryPlot	List of Loads Summary Plot displays	Flat
NyquistPlot	List of Nyquist Plot displays	Flat
Slider	List of Slider displays	Flat
StripChart	List of Strip chart displays	Flat
TppDefinitions	List of TPP parameters validated	Flat
Users	List of user defined Users	Flat
Lists	<blank>	Flat
PeaksLog	User selected peak values	Flat
SystemValues	User defined System values	Flat
ToolPositions	Internal table used for positions	Flat
ThresholdLog	Calculated thresholds	Flat
ViewQueries	Internal table of view queries	Flat
SelectionsLog	User data selections	Flat
SystemParameterDefaults	List of System Parameter Defaults	Flat
ValidationLog	Results of TPP parameter validation	Flat
DataEditLog	List of data edits performed	Flat
NullCorrections	System calculated Null corrections	Flat
FESParameters	Parameters used for the FES automation	Flat
ActiveXControlsTab	ActiveX displays on display builder tab	Flat
ActiveXDisplay	List of ActiveX displays	Flat
DataViewsDisplay	List of Data Views displays	Flat
DataGroups	User defined Data Groups	Flat
DerivativeSummaryLog	List of pEst calculated derivatives	Flat
OctaveBandDisplay	List of Octave Band displays	Flat
TestPointLog	List of completed test points	Flat
PlannedTestPoints	List of Planned test points	Flat
Maneuvers	List of pEst required maneuvers	Flat
FlightConditions	List of pEst required Flight conditions	Flat
PredictedResults	List of pEst required Predicted Results	Flat
CurrentFlightInformation2	Additional Flight information	Flat

7. Appendix B - Enumerated Data Types

7.1. IADS Data Types

IADS Data Types		
iadsInteger	0	Integer data type.
iadsDiscrete	1	Discrete data type.
iadsFloatingPoint	2	Floating point data type.
iadsLong	3	Long data type.
iadsUnsignedLong	4	Unsigned long data type.
iadsDouble	5	Double data type.
iadsAscii	6	ASCII data type.
iadsBlob	7	Binary data type

7.2. IADS Data Source Type

IadsDataSourceType		
iadsTpp	1	TPP data source
iadsDerived	2	Derived data source
iadsIap	3	Derived data source.

7.3. On/Off Enumeration

On/Off enumeration		
iadsOn	0	On setting
iadsOff	1	Off setting

7.4. Yes/No Enumeration

Yes/No enumeration		
iadsYes	0	Yes setting
iadsNo	1	No setting

7.5. Filter Algorithms

Filter algorithms		
iadsFilterNone	0	No filter algorithm
iadsButterworthFilter	1	Butterworth filter
iadsEllipticFilter	2	Elliptic filter

7.6. Filter Pass Types

Filter pass types		
iadsLowPass	1	Low pass filter
iadsHighPass	2	High pass filter
iadsBandPass	3	Band pass filter

7.7. Data Correction Methods

Data correction methods		

iadsDataCorrectionNone	0	No data correction
iadsDefaultValue	1	Default value
iadsLastValue	2	Last value

7.8. Null Corrections

Null corrections		
iadsNullCorrectionNo	0	No null correction
iadsNullCorrectionYes	1	Use null correction
iadsNullEquationInput	2	Equation input correction
iadsNullEquationResult	3	Equation result correction

7.9. Null Group Enumeration

Null group enumeration		
iadsAircraftGroup	1	Aircraft group
iadsWeaponsGroup	2	Weapons group

7.10. Spike Detection Methods

Spike detection methods		
iadsSpikeDetectionMethodNone	0	No spike detection
iadsSlopeChange	1	Slope change detection
iadsAbsoluteChange	2	Absolute change detection

7.11. IADS Compute Types

IADS compute types	Enum Value	**Description**
iadsAutoSpectrum	0	Auto spectrum compute type
iadsPsd	1	PSD compute type
iadsPhaseMagnitude	2	Phase magnitude compute type
iadsPhaseReal	3	Phase real compute type
iadsPhaseImaginary	4	Phase imaginary compute type
iadsPhaseGain	5	Phase gain compute type
IadsBode	6	Bode compute type
IadsNyquist	7	Nyquist compute type

7.12. IADS Window Types

IADS Window types	Value	**Description**
iadsWindowTypeNone	0	Default window type (none)
iadsHanning	1	Hanning window
iadsHamming	2	Hamming window
iadsBlackman	3	Blackman window
iadsKaiserBessel	4	Kaiser Bessel window
iadsRectangular	5	Rectangular window
iadsFlatTop	6	Flat Top Window

7.13. IADS Alpha

IADS Alpha	Value	Description
iadsAlphaNone	0	No alpha
iadsAlphaTwoPointZero	1	2.0 alpha
iadsAlphaTwoPointFive	2	2.5 alpha
iadsAlphaThreePointZero	3	3.0 alpha
iadsAlphaThreePointFive	4	3.5 alpha

7.14. IADS Averaging Methods

IADS Averaging Methods		
iadsAverageMethodNone	0	No averaging method
iadsAverageTime	1	Time averaging method
iadsAverageFrequency	2	Frequency averaging method

7.15. IADS Block Sizes

IADS Block Sizes (in bytes)		
iadsBlock64	64	64 byte block
iadsBlock128	128	128 byte block
iadsBlock256	256	256 byte block
iadsBlock512	512	512 byte block
iadsBlock1024	1024	1024 byte block
iadsBlock2048	2048	2048 byte block
iadsBlock4096	4096	4096 byte block
iadsBlock8192	8192	8192 byte block
iadsBlock16384	16384	16384 byte block
iadsBlock32768	32768	32768 byte block
iadsBlock65536	65536	65536 byte block

7.16. IADS Threshold Levels

IADS Threshold levels	Value	Description
iadsNoThreshold	0	
iadsWarning	1	
iadsAlarm	2	