DRAFT

SimLink User Interface Development Guidance

Prepared for

Internal

April, 2015



Office address  
Office address  
Office address

Contents

[Contents iii](#_Toc495332572)

[Acronyms and Abbreviations vii](#_Toc495332573)

[SimLink Overview 1-1](#_Toc495332574)

[1.1 SimLink Concept 1-1](#_Toc495332575)

[1.2 App Directory 1-1](#_Toc495332576)

[1.3 Datasource 1-1](#_Toc495332577)

[1.3.1 Choosing an Alternate Data Source 1-1](#_Toc495332578)

[Scenario Management 2-1](#_Toc495332579)

[2.1 Defining a Project 2-1](#_Toc495332580)

[2.1.1 Importing Network Topology 2-1](#_Toc495332581)

[2.2 Defining an Evaluation Group 2-1](#_Toc495332582)

[2.2.1 Role of Evaluation Group 2-1](#_Toc495332583)

[2.2.2 SQL 2-1](#_Toc495332584)

[2.2.3 Reference Evaluation Group 2-2](#_Toc495332585)

[2.2.4 Creating an Evaluation Group 2-2](#_Toc495332586)

[2.2.5 Viewing an Evaluation Group 2-4](#_Toc495332587)

[2.2.6 Splints 2-4](#_Toc495332588)

[2.3 Scenarios 2-4](#_Toc495332589)

[2.4 File Management 2-4](#_Toc495332590)

[2.5 Project Level Supporting Data 2-4](#_Toc495332591)

[2.5.1 Option List 2-4](#_Toc495332592)

[2.5.2 Element List 2-5](#_Toc495332593)

[2.5.3 User-Defined Functions 2-6](#_Toc495332594)

[2.5.4 Constants 2-7](#_Toc495332595)

[Simulation Platform Wrappers 3-8](#_Toc495332596)

[3.1 Platform Automation 3-8](#_Toc495332597)

[3.1.1 Derived Classes 3-8](#_Toc495332598)

[3.1.2 Initializing SimLink Classes 3-8](#_Toc495332599)

[3.1.3 Derived Class Responsibilities 3-8](#_Toc495332600)

[3.2 Network Topology 3-9](#_Toc495332601)

[3.2.1 VarType\_FK 3-9](#_Toc495332602)

[3.2.2 Dependencies 3-9](#_Toc495332603)

[In order to run Simlink with EPANET, user will require EPANET installed and the path for EPANET (e.g. C:\Program Files (x86)\EPANET2) in path environment variables. 3-9](#_Toc495332604)

[Formulation 4-10](#_Toc495332605)

[4.1 Decision Variables 4-10](#_Toc495332606)

[4.1.1 Purpose 4-10](#_Toc495332607)

[4.1.2 SQL 4-10](#_Toc495332608)

[4.1.3 Key Concepts 4-10](#_Toc495332609)

[4.1.4 User Options For Fields On Form 4-10](#_Toc495332610)

[4.1.5 Importing Decision Variables 4-13](#_Toc495332611)

[4.2 Results (Summary) 4-13](#_Toc495332612)

[4.2.1 Purpose 4-13](#_Toc495332613)

[4.2.2 Import 4-13](#_Toc495332614)

[4.2.3 User Options For Fields On Form 4-13](#_Toc495332615)

[4.3 Results (Timeseries) 4-14](#_Toc495332616)

[4.3.1 Purpose 4-14](#_Toc495332617)

[4.3.2 SQL 4-14](#_Toc495332618)

[4.3.3 User Options For Fields On Form 4-14](#_Toc495332619)

[4.3.4 CSV Import / Export 4-16](#_Toc495332620)

[4.3.5 tblExternalDataRequest, tblSupportingFileSpec and tblExternalGroup 4-16](#_Toc495332621)

[4.4 Results (Events) 4-20](#_Toc495332622)

[4.4.1 Purpose 4-20](#_Toc495332623)

[4.4.2 SQL 4-20](#_Toc495332624)

[4.4.3 User Options For Fields On Form 4-20](#_Toc495332625)

[4.4.4 CSV Import / Export 4-22](#_Toc495332626)

[4.5 Performance Variables 4-22](#_Toc495332627)

[4.5.1 Purpose 4-22](#_Toc495332628)

[4.5.2 SQL 4-22](#_Toc495332629)

[4.5.3 General Guidance 4-23](#_Toc495332630)

[4.5.4 CSV Import / Export 4-25](#_Toc495332631)

[4.5.5 Specific Guidance 4-27](#_Toc495332632)

[Running SimLink 5-28](#_Toc495332633)

[5.1 User Defined Simulations 5-28](#_Toc495332634)

[5.1.1 Staging SimLink Development for Scenario Life Cycle Tracking 5-28](#_Toc495332635)

[5.1.2 User-Defined Scenarios Grid 5-28](#_Toc495332636)

[5.1.3 Importing Scenarios 5-28](#_Toc495332637)

[5.1.4 Running User –Defined Runs 5-29](#_Toc495332638)

[5.2 Optimization 5-29](#_Toc495332639)

[5.2.1 Purpose 5-29](#_Toc495332640)

[Model Type Configuration 6-32](#_Toc495332641)

[6.1 ExtendSim 6-32](#_Toc495332642)

[6.1.1 ExtendSim Model Requirements 6-32](#_Toc495332643)

[6.1.2 Referencing fields in ExtendSim 6-32](#_Toc495332644)

[6.2 EPANET 6-33](#_Toc495332645)

[6.2.1 EPANET Model Requirements 6-33](#_Toc495332646)

[6.2.2 Referencing Input Fields 6-33](#_Toc495332647)

[6.2.3 Referencing Output Fields 6-34](#_Toc495332648)

[6.3 VISSIM 6-34](#_Toc495332649)

[6.3.1 VISSIM Model Requirements 6-34](#_Toc495332650)

[6.3.2 Referencing fields in VISSIM 6-35](#_Toc495332651)

[6.4 Cross-Model (XModel) Linkages 6-36](#_Toc495332652)

[6.5 Splints 6-36](#_Toc495332653)

[Instructions For Use 7-37](#_Toc495332654)

[7.1 Connect to a Database 7-37](#_Toc495332655)

[7.2 Create a new project 7-38](#_Toc495332656)

[7.3 Open a project 7-39](#_Toc495332657)

[7.4 Project Configuration – Element and Option Lists 7-40](#_Toc495332658)

[7.5 Evaluation Configuration – DV Editor Grid 7-42](#_Toc495332659)

[7.6 Evaluation Configuration – Result Time Series 7-43](#_Toc495332660)

[7.7 Evaluation Configuration – Event Result 7-44](#_Toc495332661)

[7.8 Evaluation Configuration – Performance 7-45](#_Toc495332662)

[Command Line Interface (CLI) 8-47](#_Toc495332663)

[8.1 Setup 8-47](#_Toc495332664)

[8.2 Run 8-47](#_Toc495332665)

[8.3 Troubleshooting 8-48](#_Toc495332666)

Acronyms and Abbreviations

CSO

GSI

MWRDGC

RPF

TARP

Section 1

# SimLink Overview

## SimLink Concept

TODO: Met to fill in the SimLink concept

Reference figure

## App Directory

C:\Program Files\SimLink- I propose this as app dir. Please let me know if there are better locations.

Is best way to store on \_simlink.\_sAppDir = ....?

Please suggest a good resolution and implement in manner that relatively low LOE

## Datasource

C:\Program Files\SimLink\db: default location for default database

(app\_dir + [\\db](file:///\\db))

### Choosing an Alternate Data Source

User should be able to navigate to alternate db location than default.

This is set on the ‘back page’ under ‘options’... Please add Options beneath Open Project.

Please add two menu items :

Please add “ Select new Access Database”- goes to browse for file filter by .mdb .accdb

“Link to SQL Server” – leave empty for now, but this allows user to enter a connection string to SQL Server database.

Reload project list following change db.

# Scenario Management

## Defining a Project

A project is the highest level container for a Simlink evaluation. Some types of data are stored at the project level, and available to all evaluation groups (see next section) within that project.

A project generally references a specific type of simulation platform (e.g. Infoworks CS, EPANET, MODFLOW). This is the general case, however for certain multi-application linked workflows (e.g. where SimClim links to EPA-SWMM) the model type must be defined at the evaluation group. This case is ignored for now.

The **simlink** class defines the core data structures and functionality associated with a project/evaluation, however these functionalities are extended in platform-specific wrapper classes, as described further in Section 3.X.

### Importing Network Topology

The user may wish to import a file associated with the project that defines the network topology. This is sometimes very useful. Please implement the following steps on the Create Project form

1. Add CheckBox for “Import Base Model”
2. If CB==true, include a browse control. Upon selection of a valid file (We will worry about extension filters later) , add placeholder call to simlink engine function (empty function- raise dialog for now).

## Defining an Evaluation Group

### Role of Evaluation Group

An evaluation group (EG) is a container for a particular evaluation. A project may have just one EG, or may have several. Distinct EG are often used when a different baseline model is needed for the same formulation, or when the goal of the analysis is distinctly different than previous analyses.

### SQL

The ‘Create Evaluation Group’ form should be quite similar to the create project form.

|  |  |
| --- | --- |
| Table | SQL |
| Option  Parent | SELECT tblEvaluationGroup.EvaluationID, tblEvaluationGroup.ProjID\_FK, tblEvaluationGroup.EvaluationLabel, tblEvaluationGroup.DateCreated, ReferenceEvalID\_FK, tblEvaluationGroup.ModelFileLocation, tblEvaluationGroup.ModelType\_ID,ScenarioID\_Baseline\_FK"  + " FROM tblEvaluationGroup" |

Fields

EvaluationID: PK

ProjID\_FK: FK to Proj

EvaluationLabel- friendly name (unique within project)

ModelFileLocation: Link- it is more common to define the model file locatin on the EG, an this overrides any value on the proj

ModelTypeID- this should be filled in on EG too, because for some workflows a proj has more than one ModelType\_ID

ScenarioID\_Baseline\_FK: Key to the basline scenario for this evaluation group. When a new EG is created, the record must first be inserted, then a scenario is inserted, then this scenario ID is updated on the EG record (as in function SimlinkUI. CreateEG

ReferenceEvalID\_FK: key to reference evaluation group (See below). In the form, all EG for selected proj should be shown in a DDL, and the user can select an EG as ReferenceEvalID\_FK

### Reference Evaluation Group

The reference evaluation group (REG\_ID) states that a given EG should use the formulation of the REG\_ID. This avoids having to re-insert DV, Result, performance definition, or the need to clone these. REG\_ID is very important. In the New Haven example, 270 is the REG\_ID for EG 271.

\_simlink. GetReferenceEvalID() retrieves the REF\_ID of the loaded EG

#### Loading Formulation

When REG\_ID\_FK is set (not -1), then the REG \_ID\_FK should be used to retrieve formulation records (DV, Result, Timeseries, performance).

#### Locked for editing

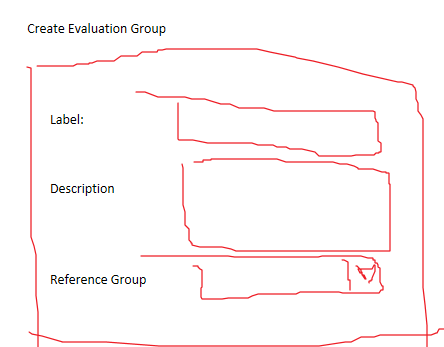
Low priority.

It seems like it would be good to lock the formulation tables for editing when REF\_ID\_FK is used- you should only be able to edit the formulation in the base EG.

### Creating an Evaluation Group

Home>>Manage Groups >> Create New Group

This should open a small, modal dialog, similar to diagram below.



#### User Options for Fields on Form – TO BE COMPLETED (Consult MET)

| **Database Field** | **Title For Grid** | **Record Display on Grid** | **User Options** | **Conditions** | **Required** |
| --- | --- | --- | --- | --- | --- |
| EvaluationID | Not Shown | Not Shown |  |  | Yes - Auto increment handled in DB |
| ProdID\_FK | Not Shown | Not Shown |  |  | Yes – Default to active ProjID |
| EvaluationLabel | Label | Value - text |  |  | Yes |
| EvaluationDescription | Description | Value - text |  |  | Yes |
| ReferenceEvalID\_FK | ReferenceGroup | DDL of all existing EG for current active ProjID. Include an option for -1 : ‘New Independent Evaluation’ |  |  | No – Should default to -1 in DB. |
| ModelFileLocation | Not Shown | Not Shown |  |  |  |
| ScenarioID\_BaseLine\_FK | Not Shown | Not Shown |  |  |  |
| ModFileKey | Not Shown | Not Shown |  |  |  |
| CohortID | Not Shown | Not Shown |  |  |  |
| CohortSQN | Not Shown | Not Shown |  |  |  |
| SimIDBaseline | Not Shown | Not Shown |  |  |  |
| IsXModel | Not Shown | Not Shown |  |  |  |
| IsModFileUserDefined | Not Shown | Not Shown |  |  |  |

### Viewing an Evaluation Group

### Splints

## Scenarios

## File Management

Fill in SimLink file system

## Project Level Supporting Data

These data are stored at project level, and always retrieved or inserted for the active project.

### Option List

Option lists are used to provide pre-defined lists of potential values to SimLink.

Example: Providing pre-defined diameter options to a decision variable

Ribbon Location: Home\\ Supporting Data \\ Option List

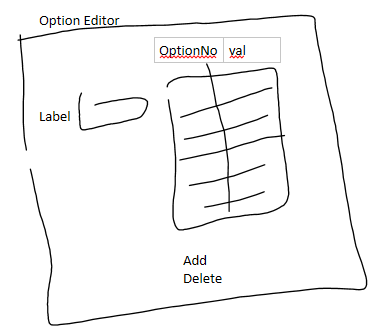
#### SQL and Table Relationships

|  |  |
| --- | --- |
| Table | SQL |
| Option  Parent | SELECT OptionID, ProjID\_FK, OptionLabel FROM tblOptionLists WHERE (((ProjID\_FK)=@ProjID)); |
| Option Detail  Child | SELECT OptionID, OptionID\_FK, OptionNo, val FROM tblOptionDetails; |

#### UI Functionality Guidance

Need to develop a small form where a user can do the following:

* Create a new option list
* Delete an option list with warning before elete)
* Add element options (number (int)/val pairs)
* Delete element options



### Element List

Element lists are lists of model elements of the same type (eg pipe, weir, pump). They enable a decision variable or results to reference numerous model elements with a single decision variable.

The ‘type’ of an element list is NOT defined for the element list- although there is a type field, this is really just for the user. The TYPE is defined by the VarType\_FK on a DV.

Also: Lists reference the linked network topology. This can be done in two fashions:

* Loosely: Only linked via VarLabel. ElementID\_FK = -1 (no explicit tie to network topology)
* Strongly: ElementID\_FK is filled in and links to a PK of network table.

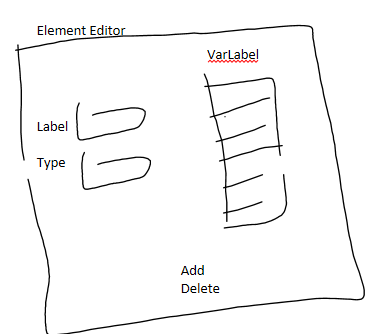
Strong references are advisable because they enable reference to other attributes on the baseline field. However, for almost all instance, a loose coupling has worked fine, and is easier to implement. Though not ideal, for now expecting a loose coupling is therefore preferred.

Ribbon Location: Home\\ Supporting Data \\ Option List

#### SQL and Table Relationships

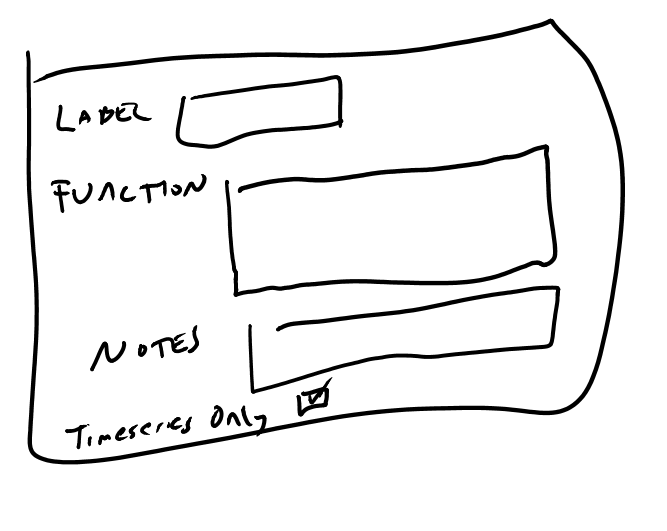
|  |  |
| --- | --- |
| Table | SQL |
| Option  Parent | SELECT ElementListID, ProjID\_FK, ElementListLabel, Type FROM tblElementLists WHERE (((ProjID\_FK)=@ProjID)); |
| Option Detail  Child | SELECT ElementListDetailID, ElementListID\_FK, ElementID\_FK, VarLabel FROM tblElementListDetails; |

#### UI Functionality Guidance



### User-Defined Functions

User-defined functions can be added for the calculation of DV or performance values (as opposed to a lookup list).

OOPS

I forgot to add Category. Please add a small edit box for Category beneath label.

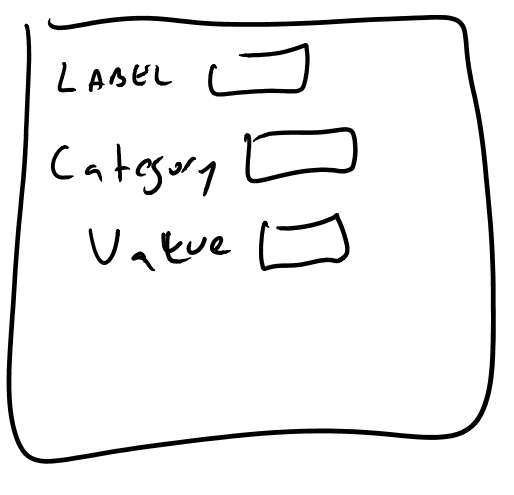
#### Function Editor (Future functionality)

Creating a viable function requires the user to follow a special syntax; this will be hard for most users to create. In the future, a function editor would allow users to create functions that reference particular SimLink data types in a more intuitive manner.

NOT FOR CURRENT IMPLEMENTATION

### Constants

The user may add constants for referencing in functions. These are tied to a PROJECT, except for pre-loaded constants which are tied to ProjID\_F=-1 (This is not relevant for this task- but is relevant when loading to a drop down list).



# Simulation Platform Wrappers

## Platform Automation

SimLink must know how to automate key aspects of the simulation life-cycle for all supported software. While the data storage is common across platforms (e.g. definition of decision variables, storage of results, definition of performance does not depend, structure wise on the platform that is used), the simulation automation is specific to that application.

### Derived Classes

Specific classes derive from simlink.cs and extend its functionality, providing the ability to modify network definition, execute simulations, and extract results. Table 3.1.1 summarizes existing platform wrappers

|  |  |
| --- | --- |
| Class | Degree of Development |
| Swmm5022.cs  EPA-SWMM | High- frequently used |
| Epanet\_link.cs  EPANET | Moderate- close to being able to apply this |
| Modflow\_link.cs  MODFLOW | Moderate- relatively simple functionality and not widely applie |
| Excel\_wrap.cs  Excel | Low- building blocks for generalized linkage to an Excel workbook representing a simulation of some type |
| ISIS2D.cs  ISIS-2D | Low- used for SimClim-ISIS linkag |

In general, most applications at this point are swmm or epa\_net.

### Initializing SimLink Classes

When a project is opened, an ActiveModelTypeID is defined associated with that project. The SimLink project should be created for that derived class, for example:

simlink \_simlink = new EPANET\_link();

simlink \_simlink = new swmm5022\_link();

### Derived Class Responsibilities

The derived class must provide specific functions that are defined as virtual in simlink.cs and are overridden in the derived classs. Some specific functions which need to be implemented include:

InitializeModelLinkage

ProcessEvaluationGroup

ProcessScenarioWRAP

InitNavigationDict

There is probably a role for interfaces in this. The required implementation for the derived platform specific wrapper needs to be defined better.

ModelTypeID codes defined below, and in tlkpUIDictionary.

\_nEvalID\_SL\_LITE = 99999; //placeholder val if no backend db used

// vals set in tlkpUIDictionary in SimLink (move to webconfig?) \_nModelTypeID\_SWMM = 1;

\_nModelTypeID\_IW = 2;

\_nModelTypeID\_MODFLOW = 4;

\_nModelTypeID\_SimClim = 8;

\_nModelTypeID\_ISIS1D = 7;

\_nModelTypeID\_ISIS2D = 9;

\_nModelTypeID\_ISIS\_FAST = 10;

\_nModelTypeID\_ExtendSim = 11;

\_nModelTypeID\_EPANET = 3;

\_nModelTypeID\_Excel = 12;

## Network Topology

TODO: Met to fill in the SimLink concept

Reference figure

### VarType\_FK

### Dependencies

### In order to run Simlink with EPANET, user will require EPANET installed and the path for EPANET (e.g. C:\Program Files (x86)\EPANET2) in path environment variables.

# Formulation

## Decision Variables

### Purpose

Decision variables allow the user to define either functions or look up list for creating new model parameters based upon a value (generally from an algorithm). Decision variables tie closely to the particular framework being modeled; they require a fine degree of granularity to enable detailed model changes for potentially large numbers of model elements.

### SQL

### Key Concepts

* Primary vs Secondary: Secondary decision variables ALWAYS reference a primary variable, and use the same DNA bit to create changes. Secondary decision variables enable consequential/contingent changes
* Option Lists: Referenced list of pre-calculated values that are used (lookup table)
* Functions- Functions may calculate the decision variable based upon the DNA bit- they may ALSO use an option list
* Element List: a list of model elements for which model changes are generated. Decision variables are always linked to an element list (even if only one model element is changed, it must be in a list – though in the future this may be changed)

### User Options For Fields On Form

| **Database Field** | **Title For Grid** | **Record Display on Grid** | **User Options** | **Conditions** | **Required** |
| --- | --- | --- | --- | --- | --- |
| DVD\_ID | Not shown | Not shown |  |  | Yes - Auto increment handled in DB |
| DV\_Label | Label | DV\_Label |  |  | Yes |
| EvaluationGroup\_FK | Not shown | Not shown |  |  | Yes – Default to active EG |
| VarType\_FK | Type | FK to required model type FieldDictionary - show FieldName | DDL of options in model type FieldDictionary with IsResult = false |  | Yes |
| DV\_Description | Not shown |  |  |  | No |
| DV\_Type | Not shown |  |  |  | No |
| Option\_FK | Option | OptionLabel from tblOptionLists | DDL link to tblOptionLists table for active ProjID. Display OptionLabel but save OptionID to database |  | Yes |
| Option\_MIN | Min Option | Value - Int |  |  | Yes |
| Option\_MAX | Max Option | Value - Int |  |  | Yes |
| GetNewValMethod | New Val Method |  |  |  |  |
| FunctionID\_FK | Function | Label from tblFunctions based on FK | DDL link to function table for active ProjID -1: not set Else: FK to tblFunction |  |  |
| FunctionArgs | Function Args | Value - text |  | Disable inputs if FunctionID\_FK = -1 | Yes if FunctionID\_FK <> -1 |
| ElementID\_FK | Elements | ElementListLabel from tblElementLists | DDL link to tblElementLists table for active ProjID. Display ElementListLable but save ElementListID to database |  | Yes |
| IsListVar | List Variable? | Boolean - checkbox | True or False from checkbox |  |  |
| SkipMinVal | Skip Min Val Option | Boolean - checkbox | True or False from checkbox |  |  |
| sqn | Not Shown |  |  |  | Default of 100 handled in DB |
| CostID\_FK | TBC (SP to check with Mason) |  |  |  | No |
| SecondaryDV\_Key | Secondary Code | Enum text | DDL with options from SecondaryDVType enum in Simlink.cs class. Integer value needs to be saved to DB but display enum text | Disable if PrimaryDV\_ID\_FK = -1 |  |
| PrimaryDV\_ID\_FK | Primary DV |  | DDL, loaded with all the DVs that are primary (PrimaryDV\_ID\_FK ==-1) for the Current RefEG. The DDL should display the labels, but populate the DVD\_ID field. Also provide an option for value ‘-1’ with label ‘Not a Secondary DV’ |  | Yes – By default set to -1 handled by DB |
| COST\_ResultID\_FK | TBC (SP to check with Mason) |  |  |  | No |
| Qualifier1 | TBC (SP to check with Mason) |  |  |  | No |
| Qual1Pos | TBC (SP to check with Mason) |  |  |  | No |
| IsSpecialCase | Special Case? | Boolean - checkbox | True or False from checkbox |  |  |
| IsTS | Is Time Series? | Boolean - checkbox | True or False from checkbox |  |  |
| XModelID\_FK | XModel ID | int | TBC (SP to Check with Mason) |  | Yes – By default set to -1 handled by DB |
| Operation | New Value Operation On Existing Value | Value - text | DDL with options from OperationType enum in Simlink.cs class |  | Yes – By default set to ‘Identity’ handled by DB |

#### Form Layout Row Filtering – same placement as other filters

Please add a drop down list has the following values (and performs the filter in parentheses):

Show all (no filter)

Primary (Primary\_DV\_ID\_FK = -1)

Secondary (Primary\_DV\_ID\_FK != -1)

#### Limited Detail and Detailed Filtering

**Limited Detail View**

DV\_Label , VarType\_FK , Option\_FK , Option\_MIN, Option\_MAX, ElementID\_FK , Operation

**Detailed View**

All fields from Section 4.1.4 that do not state ‘Not Shown’ or “TBC”

#### Option Filter Checkbox TBC (SP to check with Mason what the intention was here)

Other Filter(s): Place somewhere on form, i would prefer above grid but could be same level as ‘Save Changes’

* Option Filter (Check Box): When unchecked, Option, min, max, and operation should be hidden

### Importing Decision Variables

Sometimes there will be many decision variables, and it will be much faster to import a CSV as they can be constructed pretty efficiently in Excel. This also introduces possibility for error; hopefully this can be minimized by clear direction. This preliminary implementation of the DV import will assume the user has some familiarity with the back-end and can enter foreign keys directly (not workable long term).

**Import** (Button)- browse to DV file

Then run import script. The script creates a DV for each record and assumes the DV record is correct. There is ONE exception. IF LabelOfPrimaryIfSecondary has a value not equal to null or -1, then we need to find the dV\_ID of the DV with the LABEL (not Key, because user cannot be expected to know this key as it may have just been inserted), gets the DV\_ID, and inserts the DV\_ID into the field PrimaryDV\_ID\_FK. The purpose of this is to associate the newly imported secondary decision variable (contingent DV) with its parent/primary DV.

Message: “XXX decision variables imported successfully.” If there were errors, report “There were errors”.

**Template** (button)- user can click on this and program goes to a ‘Save as’ window and provides the user dv\_import.csv as a template to work with.

Future Enhancements:

* Unkind to require user to know the OptionID\_FK; support use of label
* Support more than one dv import file format based upon standard configurations of the import. E.g. don’t have the function info in there if irrelevant

## Results (Summary)

### Purpose

### Import

Similar to 4.1.5: see ‘result.csv’

* I changed the field names; you should just require them to be in this order

### User Options For Fields On Form

| **Database Field** | **Title For Grid** | **Record Display on Grid** | **User Options** | **Conditions** | **Required** |
| --- | --- | --- | --- | --- | --- |
| Result\_ID | Not shown | Not shown |  |  | Yes - Auto increment handled in DB |
| Result\_Label | Label | Result\_Label |  |  | Yes |
| EvaluationGroup\_FK | Not shown | Not shown |  |  | Yes – Default to active EG |
| VarResultType\_FK | Result Data Type | FK to required model type ResultsFieldDictionary - show FieldName | DDL of options in model type ResultsFieldDictionary with IsResult = true |  | Yes |
| Result\_Description | Not shown |  |  |  | No |
| ElementID\_FK | Element Model Ref (FK) | Value -int | {TODO later: DDL for this field from model element tables} |  | No |
| Element\_Label | Element Name | Value - text |  | Disable if ElementID\_FK <> -1 | Yes if ElementID\_FK = -1 |
| IsListVar | List Variable? | Boolean - checkbox | True or False from checkbox |  |  |
| ImportResultDetail | Import Result Detail | Boolean - checkbox | True or False from checkbox |  |  |

## Results (Timeseries)

### Purpose

TimeSeries specifies data that is required to be extracted from the simulation run of the model. The user requests the model element type, the data type along with other defining parameters defining the contents of the time series. The SimLink engine will then extract the TS data storing it in the database.

### SQL

### User Options For Fields On Form

| **Database Field** | **Title For Grid** | **Record Display on Grid** | **User Options** | **Conditions** | **Required** |
| --- | --- | --- | --- | --- | --- |
| ResultsTS\_ID | Not shown | Not shown |  |  | Yes - Auto increment handled in DB |
| Result\_Label | Label | Result\_Label |  |  | Yes |
| EvaluationGroup\_FK | Not shown | Not shown |  |  | Yes – Default to active EG |
| VarResultType\_FK | Result Data Type | FK to required model type ResultsFieldDictionary - show FieldName | DDL of options in model type ResultsFieldDictionary with IsResult = true |  | Yes |
| StationID\_FK | Not Shown |  |  |  | No |
| Result\_Description | Description | Value - text |  |  | Yes |
| ElementIndex | Not Shown | Not Shown |  |  | No – defaults to -1 handled in DB |
| ElementID\_FK | Element Model Ref (FK) | Value -int | {TODO later: DDL for this field from model element tables} |  | No |
| Element\_Label | Element Name | Value - text |  | Disable if ElementID\_FK <> -1 | Yes if ElementID\_FK = -1 |
| TS\_StartDate | Not Shown |  |  |  | No |
| TS\_StartHour | Not Shown |  |  |  | No |
| TS\_StartMin | Not Shown |  |  |  | No |
| TS\_Interval | Not Shown |  |  |  | No |
| TS\_Interval\_Unit | Not Shown |  |  |  | No |
| BeginPeriodNo | Start At Period | Value - int |  |  | Yes – default of 1 already implemented in database |
| SQN | Not Shown |  |  |  | No – default handled in DB |
| FunctionID\_FK | Function | Label from tblFunctions based on FK | DDL link to function table for active ProjID -1: not set Else: FK to tblFunction |  | No |
| FunctionArgs | Arguments | Value - text |  | Disable inputs if FunctionID\_FK = -1 | Yes if FunctionID\_FK <> -1 |
| IsSecondary | Is Secondary | Boolean - checkbox | True or False from checkbox |  |  |
| RefTS\_ID\_FK | Reference TS Label | FK to ResultTS - show ResultLabel | DDL of all other Result\_Label for the current EG | Disabled if IsSecondary = false | Yes if IsSecondary = true |
| RetrieveCode | RetrieveCode | Value - int | 1- Primary 2- Secondary 3 – External Data Request (Aux) requested once for each scenario 4 – External Data Request (Aux) but only performed once for EG at the start before processing any scenarios |  | Yes |
| AuxID\_FK | AuxID\_FK | Value – Int (or perhaps ExternalDataRequest table label) | Dropdown list of all values in the ExternalDataRequest table for current EG | Disabled if RetrieveCode = 1 or 2 | Yes if RetrieveCode = 3 or 4 |

#### Form Layout

The form should be laid out sort of like the Decision Variable form- a table, along with ability to enter and edit data both in a more detailed and less detailed way.

* Drop down or check box that allows user to select from limited detail or detailed view.

Data Table View: Good if two views- one with all cols, the other with select (same as limited detail view) (hide other cols)

**Limited Detail View**  (column label in parentheses)

Result\_Label (Label), VarResultsType\_FK (Result Data Type), Result\_Description (Description), Element\_Label (Element Name)

**Detailed View**

All fields from Section 4.3.3 that do not state ‘Not Shown’ or “TBC”

### CSV Import / Export

All fields should be shown (with the exception of the ID primary key) when exporting to CSV. The titles in the header line of the CSV export file should be identical to the ‘Title For Grid’. The column header should indicate whether there is a default value for the field in the database (Oledb\_Maintenance.cs contains examples of how this can be achieved). This should be retrieved dynamically using SQL for SQL server DB (maybe DAO/ADO for access DB). If an entry is missing, the default value should be applied through the database – not hardcoded in SimlinkUI.cs.

TODO - Any FK should be worked out by the UI (User should not need to know the FK, only the label as indicated on the UI)

i.e.

‘BeginPeriodNumber’ field in database should have header: **‘Start At Period (Default: 1)’**

‘FunctionID\_FK ’ field in database should have header: ‘**Function (Default: -1)**’

### tblExternalDataRequest, tblSupportingFileSpec and tblExternalGroup

Timeseries can be retrieved from an external data source using RetrieveCode = 3 or RetrieveCode = 4. (note RetrieveCode = 1 or 2 correspond to primary and secondary extractions from the simulation model)

Where RetrieveCode = 3 or 4, a foreign key linking to tblExternalDataRequest is needed to specify how to get the timeseries values.

Timeseries can be used as input to write to a text, database or webprovider through tblSupportingFileSpec.

Both tblExternalDataRequest and tblSupportingFileSpec obtain their parameters from tblExternalGroup.

A diagram of how the data is linked together is provided below.

**tblResultTS**

ResultTS\_ID

:

RetrieveCode

AuxID\_FK

**tblExternalDataRequest**

ID

Label

:

GroupID\_FK

ReturnColumnNo

ReturnColumnName

**TblSupportingFileSpec**

ID

RecordID\_FK

Source\_DataFormat

DV\_ID\_FK

GroupID\_FK

DestColumnNo

DestColumnName

**TblExternalGroup**

GroupID

EvalID\_FK

:

Params

ExternalDataCode

conn\_string

IsColIDName

IsInput

1

1

1

1

1

Many

Many

1

#### User Options For tblExternalDataRequest

| **Database Field** | **Title For Grid** | **Record Display on Grid** | **User Options** | **Conditions** | **Required** |
| --- | --- | --- | --- | --- | --- |
| ID |  | Not shown |  |  | Yes - Auto increment handled in DB |
| Label | Label | Label |  |  | Yes |
| destination\_SimlinkDataTypeCode | DataFormat | DataFormat enum | Values from enum DataFormat  DataSet = 0,  TimeSeries = 1,  Val = 2 |  | Yes |
| GroupID\_FK | ExternalGroupID | GroupID\_FK or the description field from tblExternalGroup | DDL of options in tblExternalGroup |  | Yes – this group contains the parameters for requesting this data |
| ReturnColumnNo - specifies which column 1-index (CSV or database field) to retrieve from | ReturnColumn | Value - int |  |  | Yes if IsColIDName = false for GroupID in tblExternalGroup |
| ReturnColumnName - specifies the name of the column (CSV or database field) to retrieve data from | ReturnColumnName | Value - text |  |  | Yes if IsColIDName = true for GroupID in tblExternalGroup |

#### User Options For tblSupportingFileSpec

| **Database Field** | **Title For Grid** | **Record Display on Grid** | **User Options** | **Conditions** | **Required** |
| --- | --- | --- | --- | --- | --- |
| ID |  | Not shown |  |  | Yes - Auto increment handled in DB |
| RecordID\_FK | Label | Label | DDL of options from tblResultTS |  | Yes |
| Source\_DataFormat | DataFormat | DataFormat enum | Values from enum DataFormat  DataSet = 0,  TimeSeries = 1,  Val = 2 |  | Yes |
| DVD\_ID\_FK | DV | DVD\_ID\_FK DV label from tblDV | DDL of options in tblDV with current EG |  | No – default to -1 |
| GroupID\_FK | GroupID | GroupID\_FK or the description field from tblExternalGroup | DDL of options in tblExternalGroup |  | Yes – this group contains the parameters for writing the data |
| DestColumnNo – specifies which column 1-index (CSV or database field) to write to in output | DestColumnNumber | Value - int |  |  | Yes if IsColIDName = false for GroupID in tblExternalGroup |
| DestColumnName – specifies the name of the column (CSV or database field) that will be written to | DestColumnName | Value - text |  |  | Yes if IsColIDName = true for GroupID in tblExternalGroup |

#### User Options For tblExternalGroup

| **Database Field** | **Title For Grid** | **Record Display on Grid** | **User Options** | **Conditions** | **Required** |
| --- | --- | --- | --- | --- | --- |
| GroupID | GroupID |  |  |  | Yes - Auto increment handled in DB |
| EvalID\_FK | Not shown | Not shown |  |  | Yes – Default to active EG |
| description | Description | Description |  |  | Yes |
| Params\* | params | Value – text |  |  | No – default to {empty} |
| ExternalDataCode | ExternalDataCode | Text from Enum | DDL of enum ExternalDataType. i.e. MS\_Access, WebService, XML, CSV etc |  | Yes |
| conn\_string | ConnectionString | Value - text | Varies depending on ExternalDataCode  CSV – Filename SQL or MSAccess database connection string, WebProvder - Website address |  | No |
| IsColIDName | IsColumnIDAName | Value - boolean |  |  | Yes |
| IsInput | Is an Input to Sim Model | Value - boolean | Yes is the resulting file needs to be prepared before the simulation model is run |  | Yes |

\*The individual params for the Params field can be as follows:

|  |  |  |
| --- | --- | --- |
| **params** | **Description** | **Example** |
| check\_timestamp? | Requires either a ‘y’ or ‘n’. If ‘y’, will use the start\_timestamp specified as a user input | “check\_timestamp?y,” |
| timestamp\_format? | default output timestamp "yyyy-MM-dd HH:mm". If a different format is required, this can be specified | “timestamp\_format? dd/MM/yy HH:mm,” |
| Dest\_Column\_ Is\_Sequence? | Specifies whether the column output in CSV or DB is sequentially ranked according to DestColumnNo or whether the column number is explicit | “Dest\_Column\_Is\_Sequence?n” means that timeseries data for an ExternalDataRequest with DestColumnNo = 3 is written to column 3 regardless of whether there is a column 2. |
| group\_scalar? | Scalar to apply to the timeseries data. Currently only used for writing output | “group\_scalar?0.00347222222” will multiply all timeseries values by 0.00347 |
| header\_col? (ExternalDataCode = Web provider or CSV) | Specifies the name of the parameter that takes as input the return column name. Currently only used for ‘Web Provider’ | “header\_col?**tag\_names”** should be used with pull.py as the pull.py webform requires {"**tag\_names**", "M2\_28308017\_930002\_LI\_EL\_CALC, M2\_28308017\_930003\_LI\_EL\_CALC"} as one of the dictionary elements to specify which tags to pull back |
| SQL Query (ExternalDataCode = Database and External Data Request) | SQL Query to extract from the database | “Select data from tblStaticValues” |

## Results (Events)

### Purpose

Events are processes performed on timeseries data. The user requests a type of sweep to be performed on the timeseries data, according to parameters defined in the event. The SimLink engine will then run through the TS data, and may find zero, one, or many events. They are thus very closely linked to TS data. Presently events do not have their own label, but probably one will be addd.

### SQL

|  |  |
| --- | --- |
| Table | SQL |
| tblResultEventSummary | SELECT EventSummaryID, ResultTS\_or\_Event\_ID\_FK, CategoryID\_FK, EventFunctionID, Threshold\_Inst, IsOver\_Threshold\_Inst, Threshold\_Cumulative, IsOver\_Threshold\_Cumulative, InterEvent\_Threshold  FROM tblResultTS\_EventSummary; |

### User Options For Fields On Form

Please have header row to rows in width for better column sizing.

| **Database Field** | **Title For Grid** | **Record Display on Grid** | **User Options** | **Conditions** | **Required** |
| --- | --- | --- | --- | --- | --- |
| EventSummaryID | Not shown | Not shown |  |  | Yes - Auto increment handled in DB |
| ResultTS\_or\_Event\_ID\_FK | Time Series Label | FK to tblResultTS- show tblResultTS.Result\_Label | DDL of the ResultsTS.Result\_Label for the active Evaluation Group |  | Yes |
| EvalutionGroupID\_FK | Not shown | Not shown |  |  | Yes – Default to active EG |
| CategoryID\_FK | Not shown |  |  |  | No – Default -1 handled in DB |
| Description | Not shown |  |  |  | No |
| EventFunctionID (may now be redundant as only one option available) | Calc on Time Series | Description text | obtained from enum Event\_FunctionOnTimeSeries in TimeSeries.cs (also stored in Database records for field Category = ‘Event\_FunctionOnTimeSeries’ in tlkpUI\_Dictionary)  -1 Threshold And Duration Calculations – Detect events and calculate metrics in the specified time series according to the threshold parameters |  | Yes |
| InterEvent\_Threshold | Time between Events (sec) | Value - int |  |  | No |
| Threshold\_Inst | Instantaneous Threshold | Value - double |  |  | Requires one of Threshold\_Inst or Threshold\_Cumulative |
| IsOver\_Threshold\_Inst | Trig. If Over Thres (inst.) | Boolean checkbox | True or False from checkbox |  |  |
| Threshold\_Cumulative | Cumulative Threshold | Value - double |  |  | Requires one of Threshold\_Inst or Threshold\_Cumulative |
| IsOver\_Threshold\_Cumulative | Trig. If Over Thres (Cumulative) | Boolean checkbox | True or False from checkbox |  |  |
| CalcValueInExcessOfThreshold | Calc. value in excess of threshold | Boolean - checkbox | True or False from checkbox |  |  |

### CSV Import / Export

See section 4.3.4

## Performance Variables

### Purpose

Performance variables are metrics of performance of components or the system as a whole. They may be as simple as simply repeating a result (e.g. no transformation or post-processing is required on a performance variable), or quite complicated, involving reference to the numerous predecessor tables. Examples include:

* Summing up result totals from numerous locations
* Calculating the difference between baseline and alternative conditions (any result)
* Cost
* Cost effectiveness
* Objective functions: single objective aggregation of numerous competing objectives

### SQL

|  |  |
| --- | --- |
| Table | SQL |
| tblPerformance | SELECT PerformanceID, Performance\_Label, PF\_Type, CategoryID\_FK, LinkTableCode, PF\_FunctionType, EvalID\_FK, FunctionID\_FK, IsObjective, SQN, FunctionArgs, DV\_ID\_FK, OptionID\_FK FROM tblPerformance where (EvalID\_FK = @EvalID); |
| TblPerformance (with joins) | SELECT PerformanceID, Performance\_Label, PF\_Type, CategoryID\_FK, LinkTableCode, PF\_FunctionType, EvalID\_FK, FunctionID\_FK, IsObjective, SQN, FunctionArgs, DV\_ID\_FK, OptionID\_FK  FROM (((tblPerformance LEFT JOIN tblFunctions ON tblPerformance.FunctionID\_FK = tblFunctions.FunctionID) LEFT JOIN tblOptionLists ON tblPerformance.OptionID\_FK = tblOptionLists.OptionID) LEFT JOIN tblDV ON tblPerformance.DV\_ID\_FK = tblDV.DVD\_ID) LEFT JOIN tblCategory ON tblPerformance.CategoryID\_FK = tblCategory.CategoryID; |

### General Guidance

#### User Options For Fields On Form

| **Database Field** | **Title For Grid** | **Record Display on Grid** | **User Options** | **Conditions** | **Required** |
| --- | --- | --- | --- | --- | --- |
| PerformanceID | Not shown | Not shown |  |  | Yes - Auto increment handled in DB |
| Performance\_Label | Label | Value - string |  |  | Yes |
| PF\_Type | Type | Desc text | 1=Cost , 2 = Performance |  | Yes |
| CategoryID\_FK | Category | Desc text | DDL Link to tblCategory for active ProjID  -1- not set, else FK to tblCategory |  | No |
| PF\_FunctionType | Function Type | Desc text | 1. Function 2. Lookup |  | Yes |
| LinkTableCode | Linked Data (LD) | Enum text | Options should be obtained from enum LinkedDataType in Simlink.cs (also stored in Database records for field Category = ‘LinkedDataType’ in tlkpUI\_Dictionary).  -1 Not Set  2 Result Summary  3 Timeseries Results  4 DV Option  5 Event  6 Performance | 1 is never a valid option in tblPerformance  2, 3, 4, 5, 6 available for user to select only if PF\_FunctionType = 2 | Yes – only if PF\_FunctionType = 2 |
| EvalID\_FK | Not shown | Not shown |  |  | Yes – Default to active EG |
| FunctionID\_FK | Function | Label from tblFunctions based on FK | DDL link to function table for active ProjID -1: not set Else: FK to tblFunction | Disable options if PF\_FunctionType = 2 | Only if PF\_FunctionType = 1 |
| Description | Not shown |  |  |  | No |
| FunctionArgs | Arguments |  |  | Disable options if FunctionID\_FK = -1 | Only if PF\_FunctionType = 1 |
| IsObjective | Objective | Boolean checkbox | True or False from checkbox |  |  |
| DV\_ID\_FK | DV Key | DV\_Label from DVD\_ID FK | DDL of DVD\_ID and DV\_Label for the active Evaluation Group (or ref EG if applicable) from tblDV | Disable options if LinkTableCode **does not equal** 4 | Only if LinkTableCode = 4 |
| OptionID\_FK | Options | OptionLabel from OptionID FK | DDL of DVD\_ID and DV\_Label for active ProjID from tblOptionList | Disable options if LinkTableCode **does not equal** 4 | Only if LinkTableCode = 4 |
| SQN | Sequence | Value - Int |  |  | Yes |
| ApplyThreshold | Apply Threshold | Boolean checkbox |  |  | Yes |
| Threshold | Threshold Value | Value - double |  | Disable if ApplyThreshold is false | Yes if ApplyThreshold is true. |
| IsOver\_Threshold | Trig. If Over Thres | Boolean checkbox | True or False from checkbox | Disable if ApplyThreshold is false | Yes if ApplyThreshold is true. |
| ResultFunctionKey | Function on Linked Data | Enum text | DDL should be obtained from enum Perf\_FunctionOnLinkedData in Simlink.cs (also stored in Database records for field Category = ‘Perf\_FunctionOnLinkedData’ in tlkpUI\_Dictionary).  -1 Not Set  1 Sum  2 Min  3 Max  4 Count | Disable options if PF\_FunctionType **equals** 1  Disable options if LinkTableCode **equals** 4 | **Required** if PF\_FunctionType = 2  **NOT REQUIRED** if LinkTableCode **equals** 4 OR PF\_FunctionType = 1 |
| ComponentApplyThreshold | Apply Thres (Linked Values) | Boolean checkbox | True or False from checkbox |  |  |
| ComponentThreshold | Threshold (Linked Values) | Value - double |  | Disable if ComponentApplyThreshold is false | No |
| ComponentIsOver\_Threshold | Trig. If Over Thres (Linked Values) | Boolean checkbox | True or False from checkbox | Disable if ComponentApplyThreshold is false | No |

#### Form Layout

The form should be laid out sort of like the Decision Variable form- a table, along with ability to enter and edit data both in a more detailed and less detailed way.

* Drop down or check box that allows user to select from limited detail or detailed view.

Data Table View: Good if two views- one with all cols, the other with select (same as limited detail view) (hide other cols)

**Limited Detail View**  (column label in parentheses)

Performance\_Label (Label), PF\_Type (Type), CategoryID\_FK (Category), LinkTableCode (Linked Data), PF Function Type (Function Type), ApplyThreshold (Apply Threshold)

**Detailed View**

All fields from Section 4.5.3.1.

### CSV Import / Export

See section 4.3.4

#### Performance Cross Reference Table

The performance table does not link directly to records in other tables – there is no place for something like a PK to another table. All links to other tables are held in tblPerformance\_ResultXREF. This is because there is often, though not always, more than one record to be referenced for a given field.

ApplyThreshold, Threshold and IsOver\_Threshold can be specified per record in tblPerformance\_ResultXREF if required. If any of these values are NULL, the ComponentApplyThreshold, ComponentThreshold and ComponentIsOver\_Threshold from the corresponding row in tblPerformance are used. TODO: Include this functionality on the UI form for Linked Records List

For any performance record, a user should be able to link to a modular form that helps them enter one or many IDs of the proper type.

On the 'performance' tab when a record is being entered suggest a 'Linked Records' button is placed on the expanded record. This need not be shown when records are in compressed format but should be shown in the both the 'Limited' and 'Detailed' forms of the record.

The 'Linked Records' button should be disabled until the 'Linked Data' field is selected as the Linked Records will depend on the 'Linked Data' selected by the user.

The 'Linked Records' button should bring up the 'Linked Records List' box where a user can view / add / delete Linked Records for the Performance ID record selected. {LinkedData Name} on the form should be populated with the user selected 'Linked Data' DDL

Suggest that the expanded record display and pop out form should be along the lines of the following screenshots.



The DDL, the shown table, and the 'add and delete' buttons in the form can link to the following functions in SimlinkUI.cs.

|  |  |
| --- | --- |
| Features | Function / Procedure |
| Linked Record Drop Down List on pop out form | Function in the SimlinkUI:  GetLinkedRecordsLookUp(int nLinkTableCode, int nEvalID) |
| Linked Record table (Id and Linked Record Label) on pop out form | Function in the SimlinkUI:  GetLinkedRecordDetails(int nPerfID) |
| The ‘Add’ button | PerformanceCreateLinkedRecord(int nPerfID, int nLinkDataCode, int nLinkedRecordID) |
| The ‘Delete’ button | PerformanceDeleteLinkedRecord(int nPerfID, int nLinkDataCode, int nLinkedRecordID) |

### Specific Guidance

# Running SimLink

## User Defined Simulations

User-defined simulations are a useful way for users to selectively explore the decision space, useful for: testing, running specific solutions of interest, defining seed solutions for the optimization. SimLink is geared towards a flexible, customizable interaction between the user and simulation automation- meaning that the user may wish to be able to control characteristics of the simulation execution. For instance, some simulations may execute over long periods of time after SimLink is closed; the user must be able to finalize results processing and performance definition after a period of execution.

### Staging SimLink Development for Scenario Life Cycle Tracking

* Stage 1: User manages tracking ‘stage’ of life cycle in between discontinuous sessions (SimLink manages during a process)
* Stage 2: SimLink syncs scenario status in between discontinuous sessions.

UI support for Stage 1 will be a big improvement and is the current priority.

### User-Defined Scenarios Grid

The grid should contain the following columns (row 1, with sample data in row 2 and column names in row 3):

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ScenarioID | ScenarioLabel | DNA | HasBeenRun | ScenStart | ScenEnd | DateEvaluated |
| 8480 | XXXXX | 3.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1 | TRUE | 20 | 20 |  |
| ID | Label | DNA | Has Been Run | Start | End | Date Evaluated |

While IDs are not generally shown to the user, this view of the scenario will display them.

#### Launching the User-Defined Scenario Grid

This grid is distinctly different than the formulation grid, and should not be loaded in the same tabbed panel.

Add a ‘Scenario’ grouping to the ‘Home’ ribbon, and add a menu item to it called “Define runs” – this will open the gird.

#### Scenario Life-Cycle

Simlink offers the user fine control over how a scenario is executed. Stages of a scenario ‘life cycle’ are coded as numbers. For now, the user will need to define these numbers.

*nScenLCExist = 1;*

*nScenLCModElementExist = 2;*

*nScenLCBaselineFileSetup = 3;*

*nScenLCBaselineModified = 4;*

*nScenLCModelExecuted = 5;*

*nScenLCModelResultsRead = 6;*

*nScenLModelResultsTS\_Read = 10;*

*nScenResultTS\_Operations = 12;*

*nScenDefineEvents = 15;*

*nScenEventAggregation = 15;*

*nScenLCSecondaryProcessing = 20;*

### Importing Scenarios

It is too laborious to enter the number of scenarios manually (though user should have option, similar to DV grid)- there may be dozens or hundreds.

Therefore, a simple import is required for the user to import scenarios. Please see accompanying CVS ‘scenario.csv’ for the format to be supported. A record should be inserted for row in the .csv. Similar to the DV import (4.1.5), an option to export a template for the user would be nice.

### Running User –Defined Runs

#### Run all – please add to Home\Scenario\Run scenarios (menu item)- please add an empty handler, which will call a simlink function. Note: this only runs scenarios where HasBeenRun = false.

* + **Run single**: Context menu on grid row, ‘Run Scenario’ which runs that individual scenario (again - please add an empty handler. The menu item should only be enabled if ‘HasBeenRun’ = false

At present, these commands will launch new processes, and the user will need to collect results later.

## Optimization

### Purpose

* + 1. **Decision Variables**
       1. Purpose:
       2. SQL:
       3. General Guidance
       4. Specifics and additional details
          1. Secondary Decision Variables: I need to think about this and fill it in.
    2. Results: Several types of results may be defined, depending on exactly what the user needs.
       1. **Results Summary**
          1. Purpose: This table stores single point summary information linked to a single model element. At the present time, lists of result vars is not supported (unlike DV) which supports lists
          2. SQL:

SELECT tblResultVar.Result\_ID, tblResultVar.Result\_Label, tblResultVar.VarResultType\_FK, tblResultVar.Result\_Description, tblResultVar.ElementID\_FK, tblResultVar.Element\_Label

FROM tblResultVar where (EvaluationGroup\_FK = @ActiveEvalID);

* 1. General Guidance
     1. Usage and labeling in grid
        1. Result\_ID- UID- not shown
        2. Result\_Label: Name
        3. VarResultType\_FK: Field
           1. This is to be filled by drop down, via tlkpSWMMResults\_FieldDictionary.ResultsFieldID as explained further below
        4. Result\_Description: Description
        5. ElementID\_FK: Element ID
        6. Element\_Label: Element Name
  2. Specifics and Additional Details
     1. Editing/Forms: ResultVar is straightforward enough
     2. VarResultType\_FK: Like elsewhere in SimLink, this variable defines what the stored element actually does
        1. SWMM
           1. SQL:

SELECT tlkpSWMMResults\_FieldDictionary.ResultsFieldID, tlkpSWMMResults\_TableDictionary.TableName, tlkpSWMMResults\_FieldDictionary.FieldName

FROM tlkpSWMMResults\_FieldDictionary INNER JOIN tlkpSWMMResults\_TableDictionary ON tlkpSWMMResults\_FieldDictionary.TableID\_FK = tlkpSWMMResults\_TableDictionary.ResultTableID WHERE (((tlkpSWMMResults\_FieldDictionary.IsOutFileVar)=False));

* 1. Notes: tlkpSWMMResults\_FieldDictionary.ResultsFieldID is what is stored in VarType\_FK
  2. Filtering: There are lots of values returned from the above query. It would be ideal if these could be displayed, in the grid, in a two column dropdown (with the ID hidden), and the user could filter on the tablename)
  3. EPANET
  4. **Results Timeseries**
     1. Purpose: This grid identifies the locations where the user would like to pull back results from the simulation
     2. SQL:

SELECT tblResultTS.ResultTS\_ID, tblResultTS.Result\_Label, tblResultTS.VarResultType\_FK, tblResultTS.Result\_Description, tblResultTS.ElementID\_FK, tblResultTS.ElementIndex, tblResultTS.Element\_Label, tblResultTS.EvaluationGroup\_FK

FROM tblResultTS

WHERE (((tblResultTS.EvaluationGroup\_FK)=[Param]));

* 1. General Guidance
     1. Usage and labeling in grid
        1. ResultTS\_ID- not shown, UID
        2. Result\_Label: Name
        3. VarResultType\_FK: Field
           1. This is to be filled by drop down, via tlkpSWMMResults\_FieldDictionary.ResultsFieldID as explained further below
        4. Result\_Description: Description
        5. ElementID\_FK: Element ID
        6. Element\_Label: Element Name

* 1. Specifics and additional guidance
     1. Editing/Forms: The version shown above is the simple version. Similar to DV setup, there is also a more complicated version.
        1. Detailed Result TS: to be added at later date
     2. VarResultType\_FK: This varies by \_nActiveModelTypeID; for now just set up for SWMM
        1. SWMM:

SELECT tlkpSWMMResults\_FieldDictionary.ResultsFieldID, tlkpSWMMResults\_FieldDictionary.FeatureType, tlkpSWMMResults\_FieldDictionary.FieldName

FROM tlkpSWMMResults\_FieldDictionary

WHERE (((tlkpSWMMResults\_FieldDictionary.IsOutFileVar)=True));



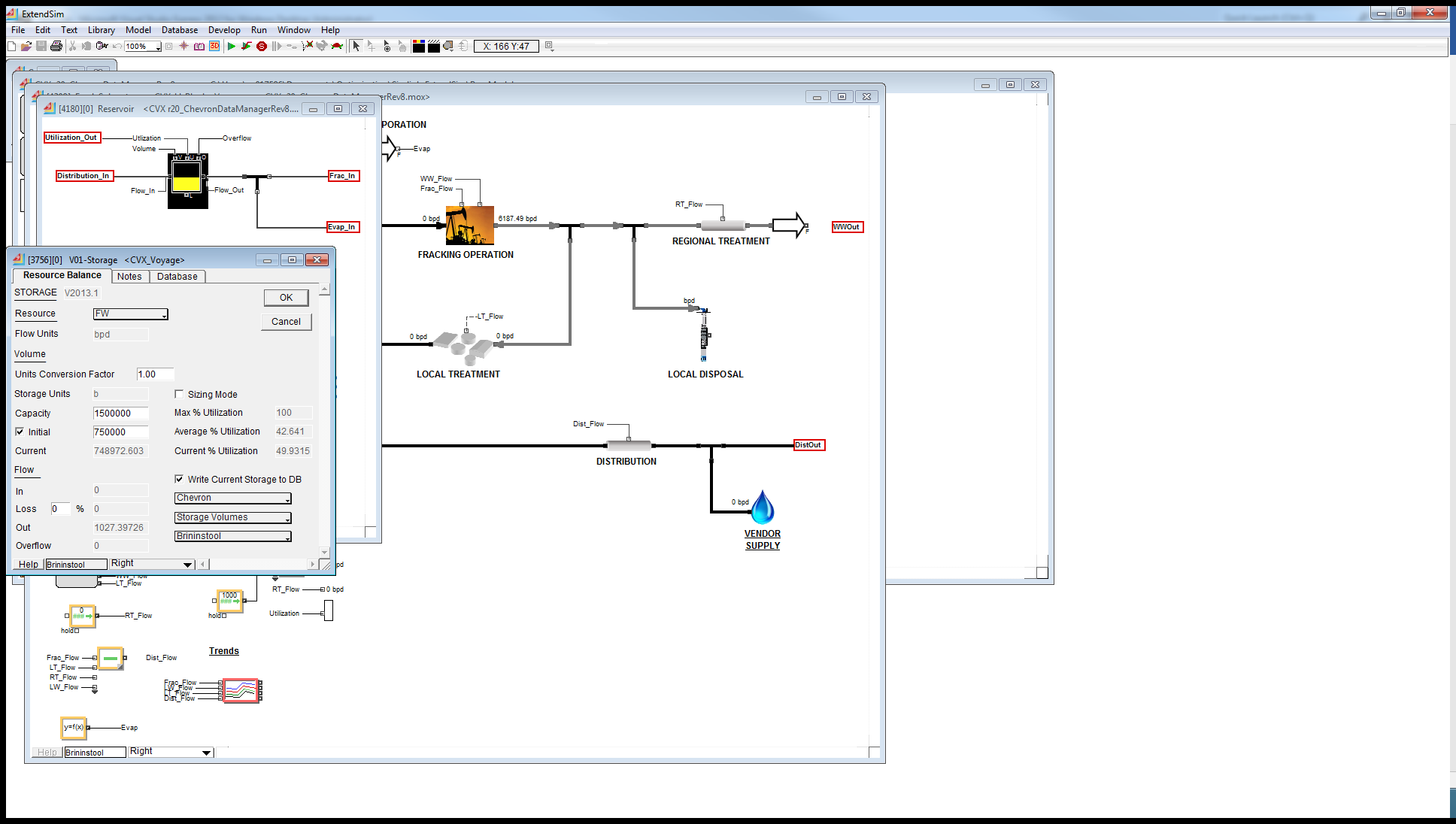
* 1. Events
  2. Performance

# Model Type Configuration

## ExtendSim

### ExtendSim Model Requirements

1. All block objects where the properties will be referenced (set or get) require a ‘Block Label’ identifier. Block labels for different block types can be identical (TBC?)



1. Time series data extraction can only be achieved through explicitly storing each simulation time step in a database table.

### Referencing fields in ExtendSim

tlkpExtend\_StructureDictionary

|  |  |
| --- | --- |
| **StructureID** | **StructureName** |
| 1 | Database |
| 2 | Block |

The syntax required for referencing an element is as follows

|  |  |  |  |
| --- | --- | --- | --- |
| **Structure Type** | **Syntax Required** | **Example** | **Explanation** |
| Database | DatabaseName; TableName; ReturnFieldName; [RecordFieldName\*]; [Record\*];  \*If RecordFieldName and Record are omitted, the entire column array will be returned | Chevron;Storage Volumes;Hayhurst SE;  OR  Chevron;Storage Data;Scenario Capacity;ID;Jumping Springs; | ReturnFieldName - the column with values to be returned  RecordFieldName [Optional] – the column of the Record to be looked up  Record [Optional] – The value in the RecordFieldName for the row to be referenced |
| Block | Block Name; Block Label; Block Property | Statistic; Hayhurst SE; Q\_Avg; | Block Name – the type of block  Block Label - the name of the specific block object in the bottom left of the block dialog.  Block Property – the name of the property to be referenced |

The following tables contain fields which require referencing ExtendSim Objects

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Simlink Table** | **Simlink Field** | **StructureID\_FK specification location** | **Can Accept Database Format** | **Can Accept Block Format** |
| tblElementListDetails | VarLabel | tblDV – VarType\_FK | ✓ - single value only | ✓ |
| tblResultVar | Element\_Label | tblResultVar – VarResultType\_FK | ✓ - single value only | ✓ |
| tblResultTS | Element\_Label | tblResultTS – VarResultType\_FK | ✓ | 🗶 |

## EPANET

### EPANET Model Requirements

* All new elements must exist in the model (can be active or inactive)
* All new controls and rules must exist in the model (can be commented out or left as is)
* Nothing other than that

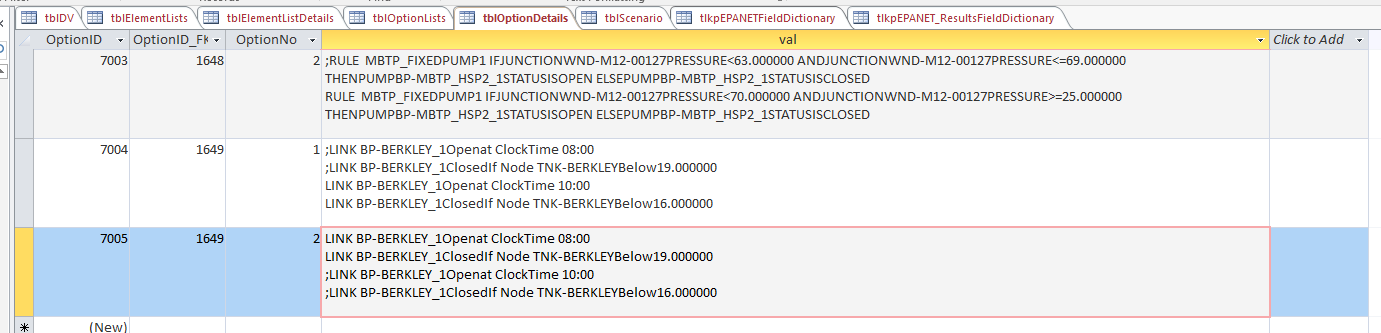
### Referencing Input Fields

.inp text file editing – field columns for all elements. Database code fields in tlkpEPANETFieldDictionary and tlkpEPANETTableDictionary allowing easy access to node and link parameters.

#### Controls and Rules

One exception is when using controls and rules in the decision variables. These must be set up as follows:

* tblDV VarType\_FK set to 359 (rule) or 360 (control)
* tblOptionDetails val field must contain the rules or controls as they need to appear in the model if that alternative option is selected. Multiple lines can be included at one time. It is recommended that all controls relating to the one facility are listed as one option with the required lines commented out to avoid the necessity for exclusivity. i.e.



* tblElementListDetails must be non-null and it is recommended to use the keyword RULE and CONTROL for the VarLabel field in this table. Multiple DVs for controls can link to the same tblElementLists for these dummy elements
* The controls and rules must exist in the respective sections with the same wording duplicated in the tblOptionDetails val field. Variations in spaces / tabs are accounted for however newline characters must be in similar locations.

#### Tanks

Requires use of a secondary DV for the input pipe to the tank. When a zero sized tank, requires this pipe is shut therefore will perform as required as long as the input pipe is a secondary DV linked to the primary DV tank storage

### Referencing Output Fields

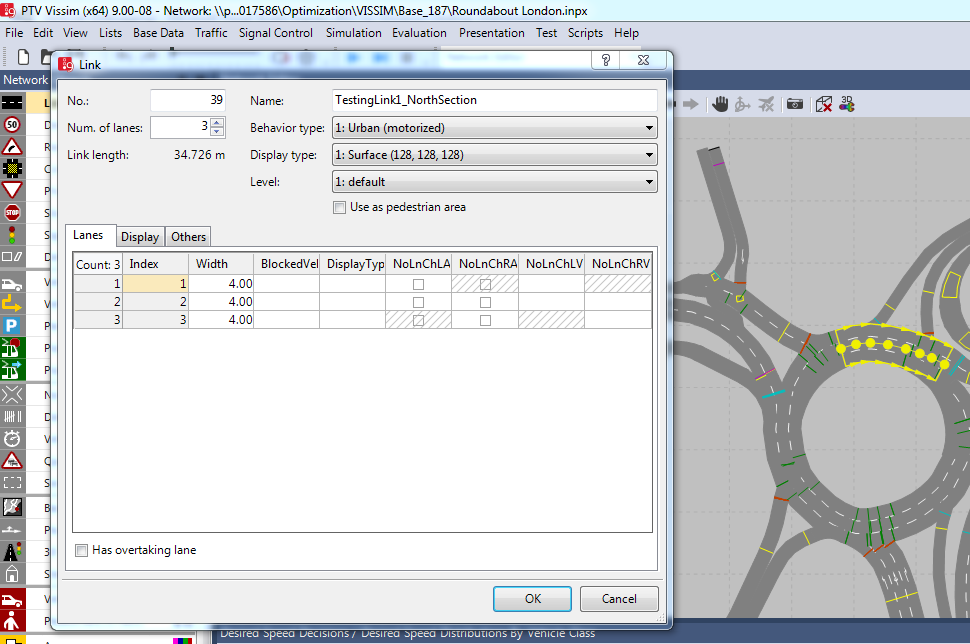
Either through dll or .OUT file. Database contains a list of codes which can be extracted for

## VISSIM

### VISSIM Model Requirements

1. All objects where the properties will be referenced (set or get) will contain an identifier that is unique for its element type. Some items are sub elements of the parent element such as lane items relating to a link as per screenshot. This can be referenced using a hyphenated unique ID ie 39-1 for lane 1 of Unique ID 39. See VISSIM 🡪 Help 🡪 COM Help for a list of all attributes and which can be modified / read or need to be referenced.

Element type LinkEvalSegment has the option of extracting values per lane and per segment in the link. A link Eval segment can both a lane and segment specified in the Element\_Label field. If they are omitted, they are assumed to be 1. i.e. ‘5-2-3’ refers to lane 2 and segment #3 of Link #5, ’10-2’ refers to lane 2 and segment 1 of Link #10

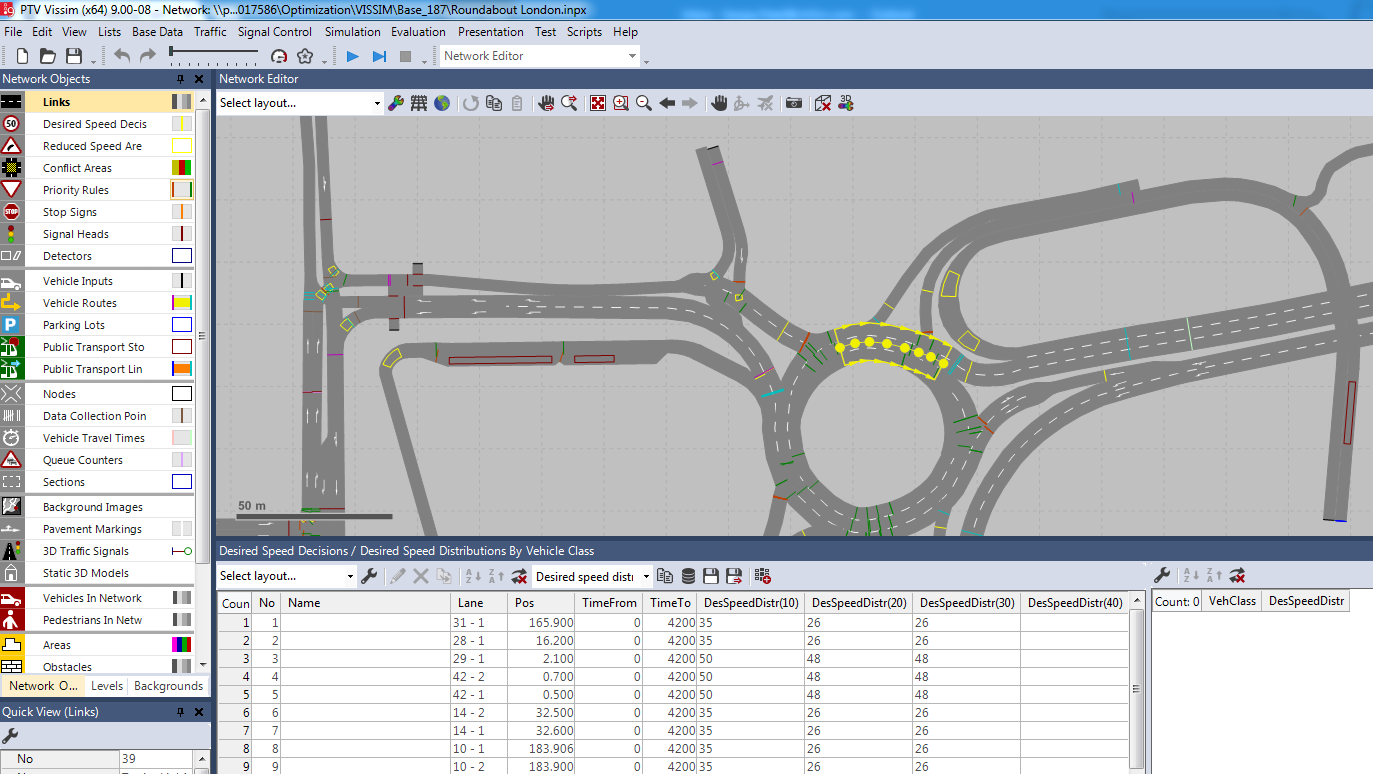


1. Time series data extraction can be achieved for any element. The interval at which time series results are retrieved is determined by the “Break At:” parameter in the Simulation Parameters setup.

### Referencing fields in VISSIM

Use VarTypeID for DVs from tlkpModelAttribute (this will soon become the normal for all model types – SP 10-Jul-2017 – see Simlink issue #35). This has only been populated with a subset of the possible attributes for a demo case but will need to be loaded with additional attributes as they are needed. Each attribute has a section ID indicating which element the attribute belongs to. Note that some attributes are part of the sub element in which case the sub element will need to be referenced in tblElementLists VarLabel.

| **ID** | **ModelTypeID\_FK** | **SectionID\_FK** | **FieldName** | **FieldAlias** | **FieldINP\_ColNo** | **FieldINP\_RowNo** | **FieldAPI\_Code** | **FieldClass** | **IsResult** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 200 | 14 | 2 | AssumSpeedOncom | AssumSpeedOncom | -1 | -1 | -1 | -1 | 0 |
| 400 | 14 | 4 | DesSpeedDistr(10) | DesSpeedDistr - Car | -1 | -1 | -1 | -1 | 0 |
| 401 | 14 | 4 | DesSpeedDistr(20) | DesSpeedDistr - HGV | -1 | -1 | -1 | -1 | 0 |
| 402 | 14 | 4 | DesSpeedDistr(30) | DesSpeedDistr - Bus | -1 | -1 | -1 | -1 | 0 |
| 600 | 14 | 6 | MinGapTime | MinGapTime | -1 | -1 | -1 | -1 | 0 |
| 601 | 14 | 6 | MinHdwy | MinHeadWay | -1 | -1 | -1 | -1 | 0 |



The following tables contain fields which require referencing VISSIM Objects

|  |  |  |
| --- | --- | --- |
| **Simlink Table** | **Simlink Field** | **ID from tlkpModelAttribute specification location** |
| tblElementListDetails | VarLabel | tblDV – VarType\_FK |
| tblResultVar | Element\_Label | tblResultVar – VarResultType\_FK |
| tblResultTS | Element\_Label | tblResultTS – VarResultType\_FK |

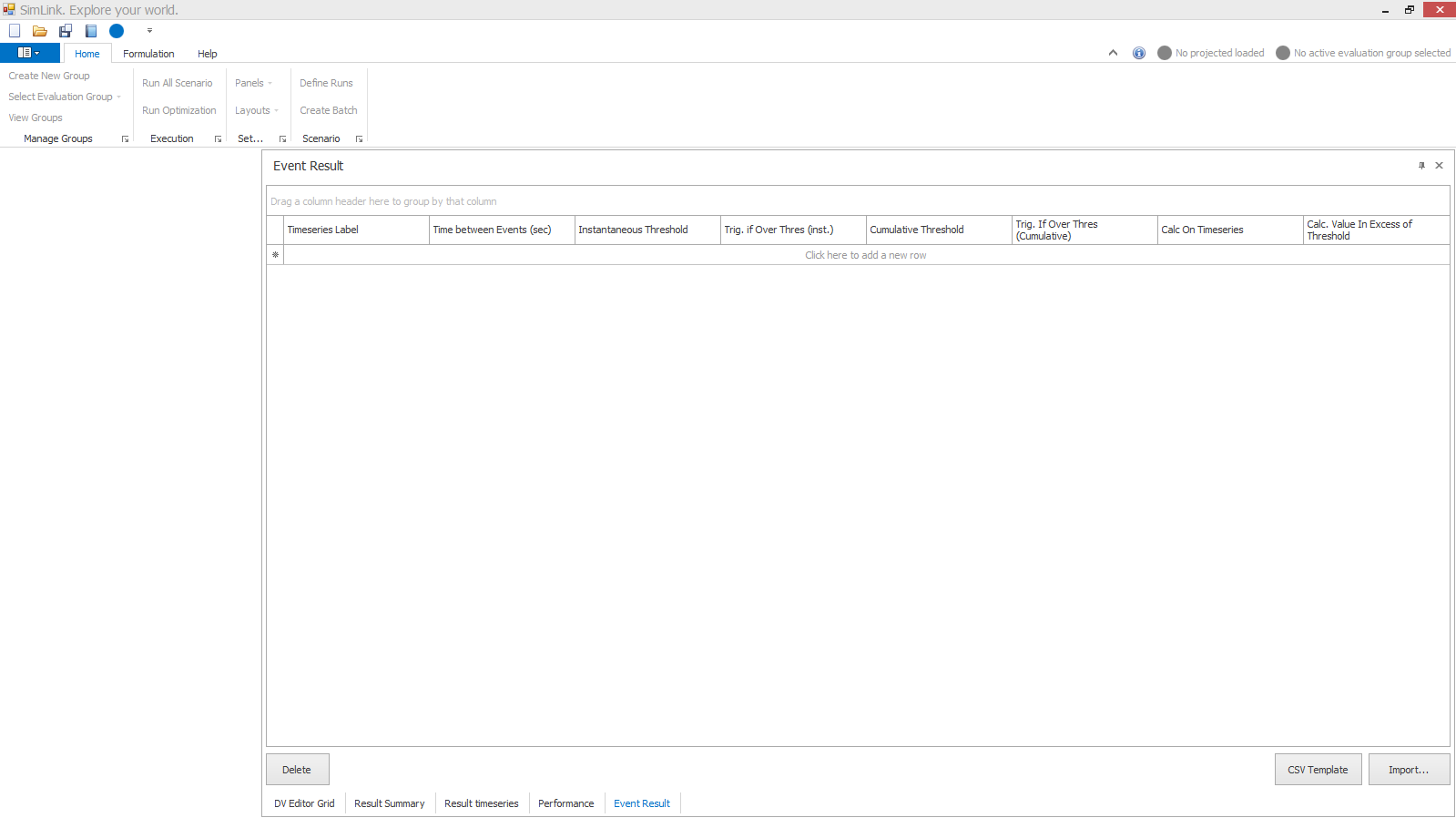
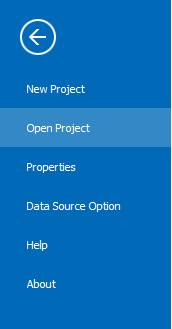
## Cross-Model (XModel) Linkages

## Splints

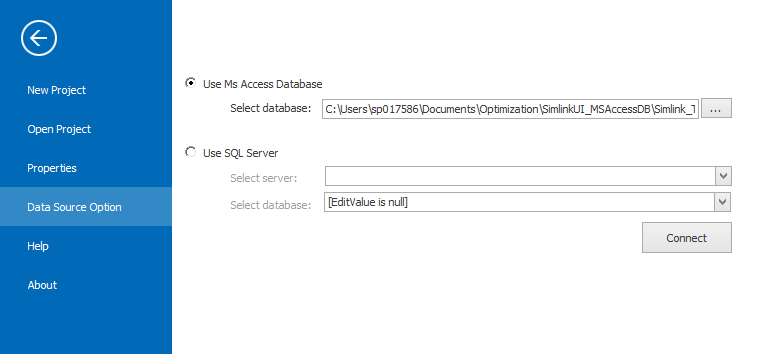
# Instructions For Use

## Connect to a Database

* Click the file button and select ‘Data Source Option’

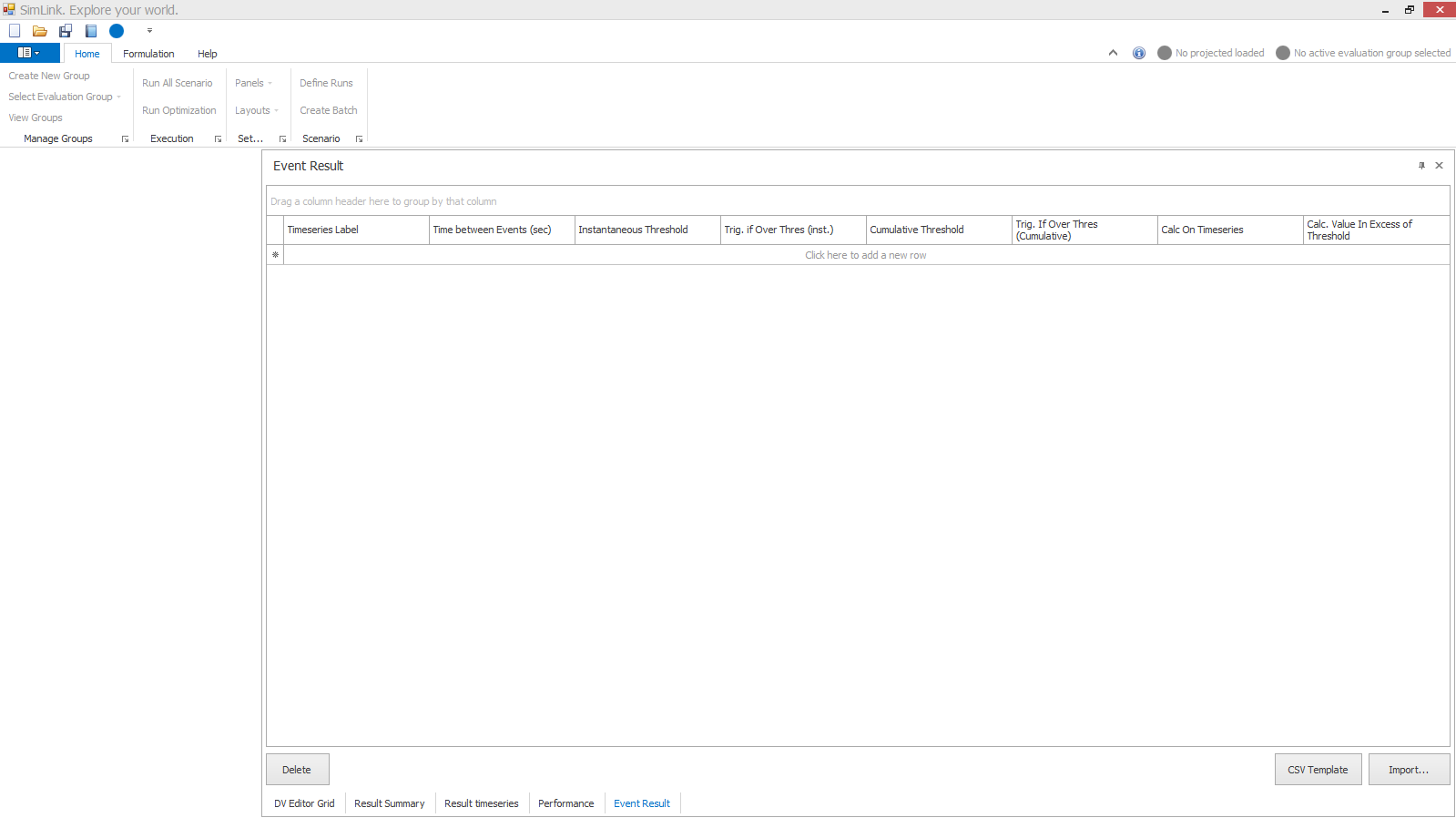
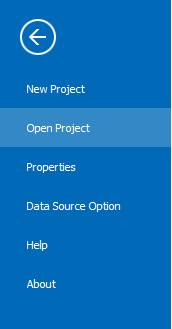
 

* Specify the database location through either MS Access or SQL Server
  + Microsoft Access database (I believe this is only compatible with .mdb files (i.e. MS Access 1997-2003 format)
    - Choose the file location
  + SQL Server
    - select server from drop down list OR type the full server location
    - select the database
* Click ‘Connect’ to ensure the connection was successful

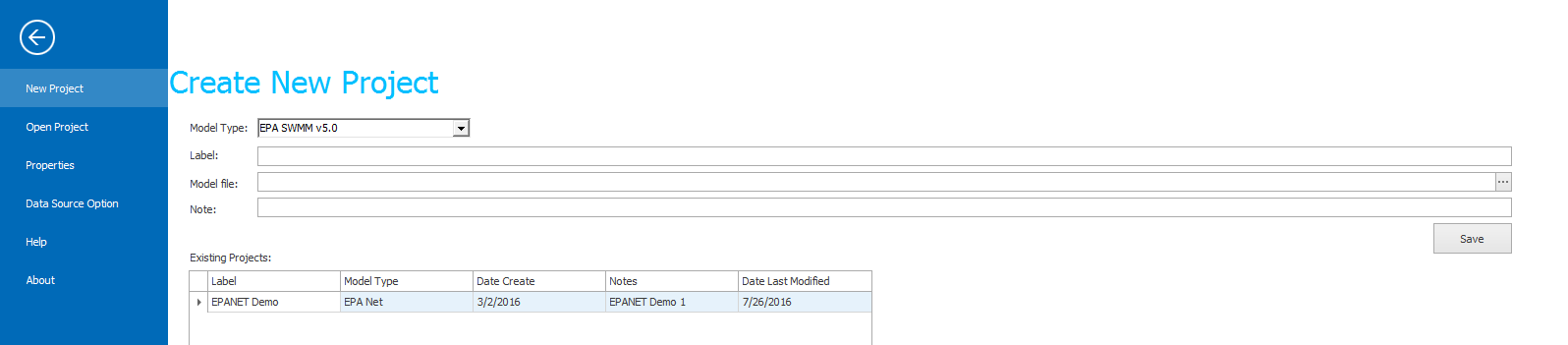


## Create a new project

* Click the file button and select ‘New Project’

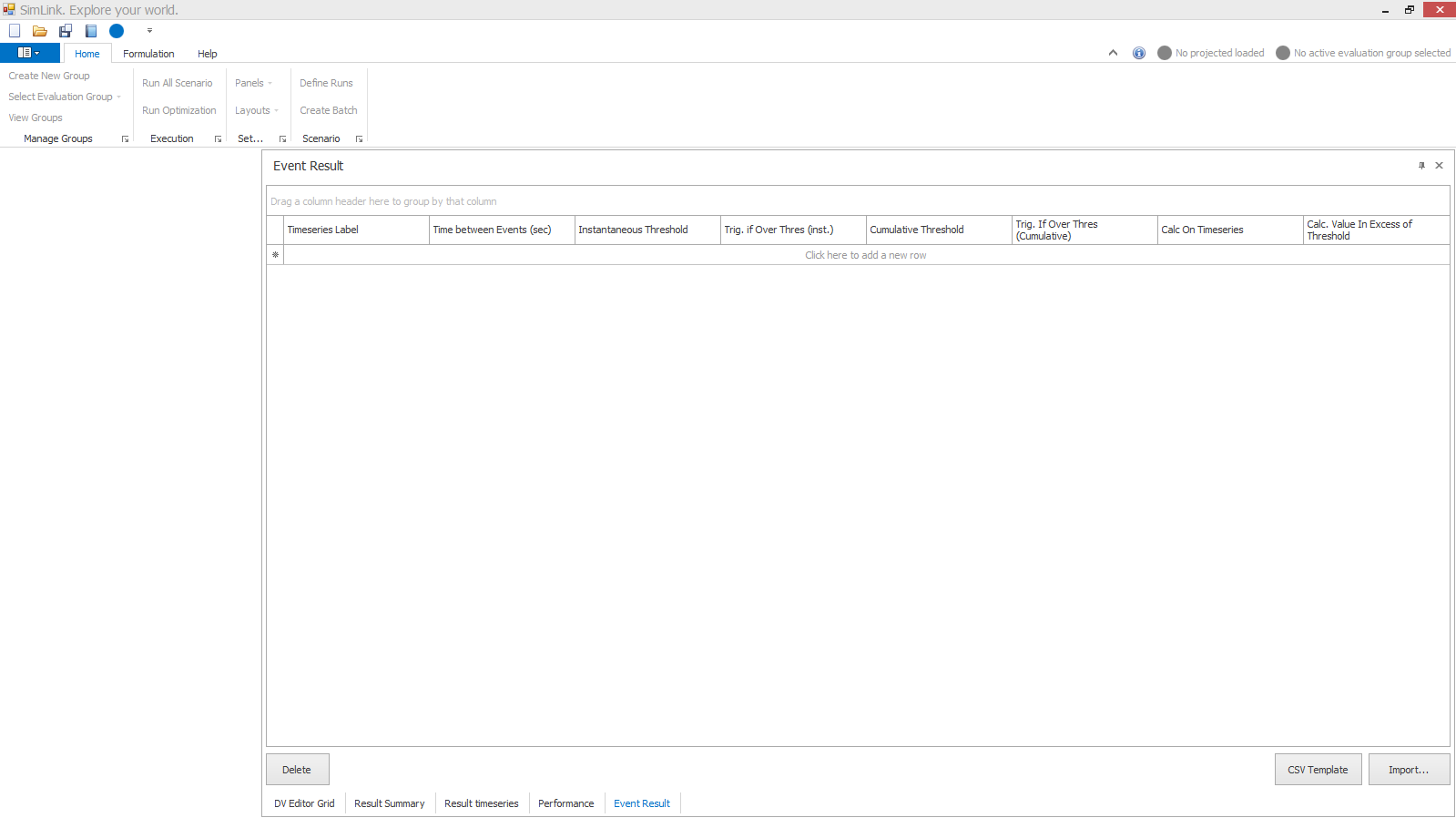
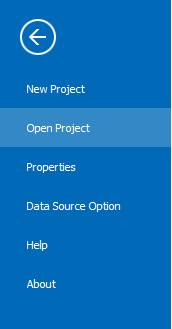
 

* Model Type: Select the model type: EPA SWMM v5.0, EPANET etc
* Label: Enter a name for the project
* Model File: Choose the model file location: this is the base model that all other scenarios will stem from
* Note: Enter any additional notes
* Click ‘Save’

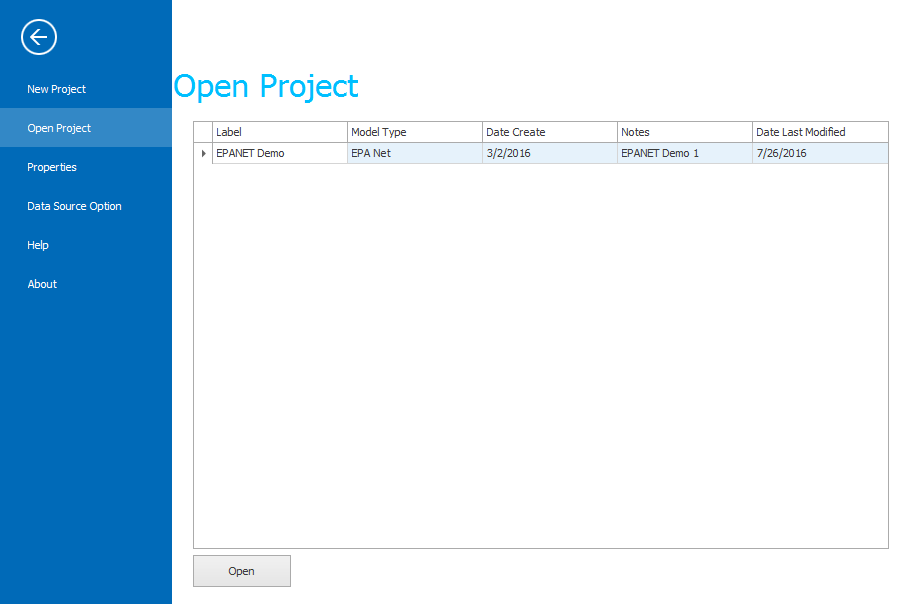


## Open a project

* Click the file button and select ‘Open Project’

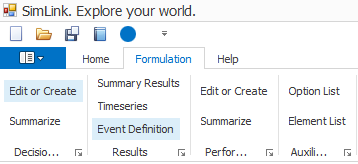
 

* Highlight the required project and click ‘Open’

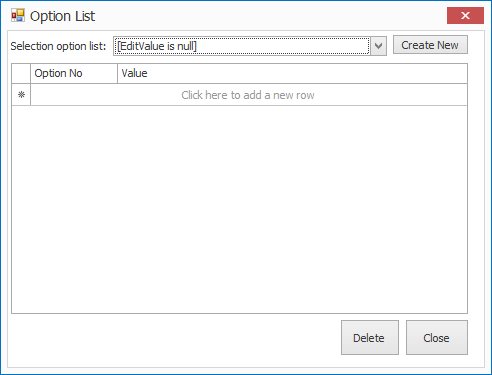


## Project Configuration – Element and Option Lists

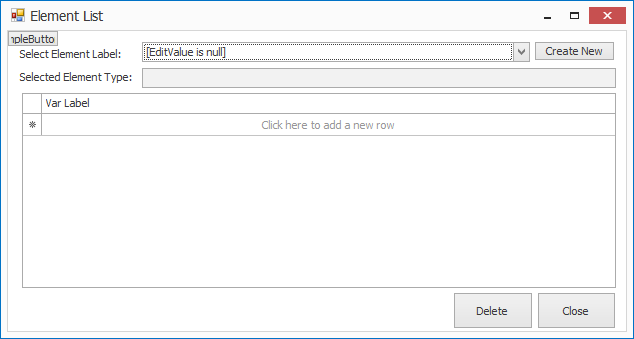
* Option and Element Lists can be configured through the ‘Formulation’ tab in the ribbon and then selecting ‘Option List’ or ‘Element List’



* Option Lists – provide option values for the decision variables
  + Create a new option list through the ‘Create New’ button (an entire option list cannot be deleted through the UI – do this through the database)
  + Add option values to an option list by clicking ‘Click here to add a new row’
  + Delete option values using the delete button

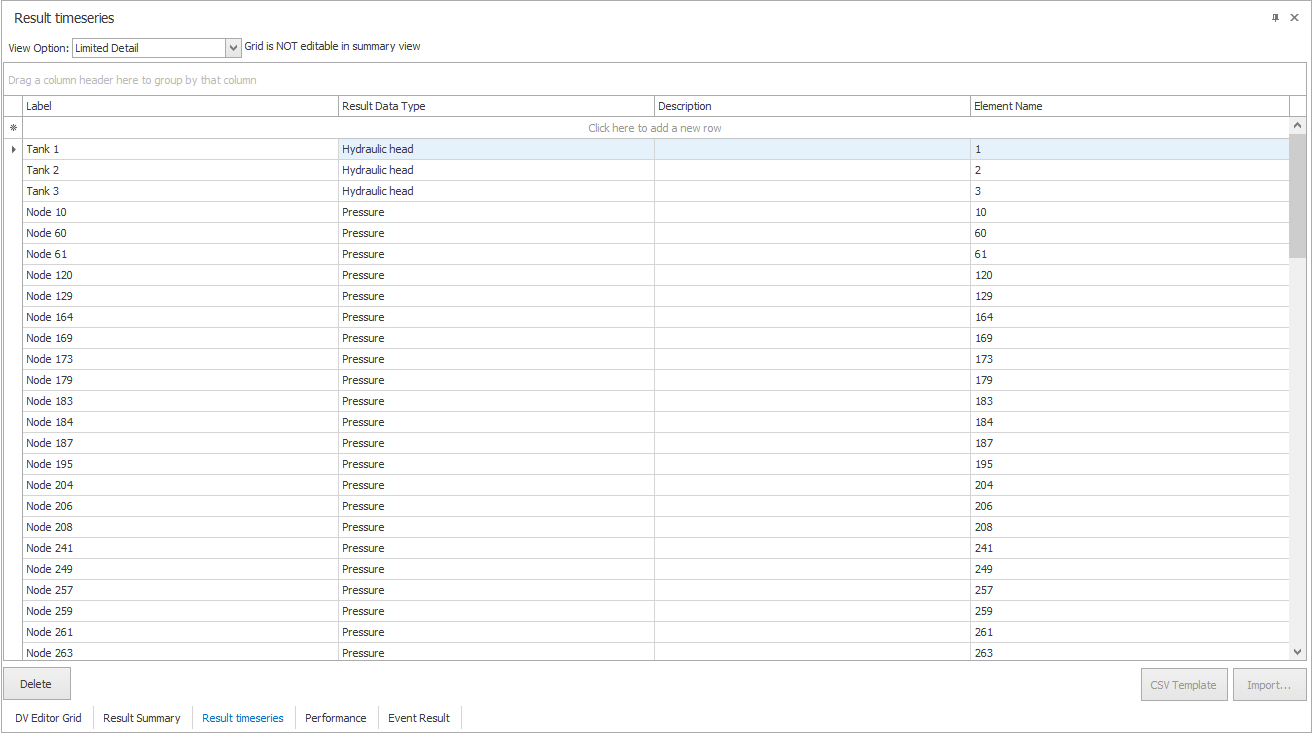


* Element Lists – specifying the names of elements in the model that can be assigned to abstract decision variables
  + Create a new element list through the ‘Create New’ button (an entire element list cannot be deleted through the UI – do this through the database)
  + Add element labels to an element list by clicking ‘Click here to add a new row’
  + Delete an element label from an element list using the delete button



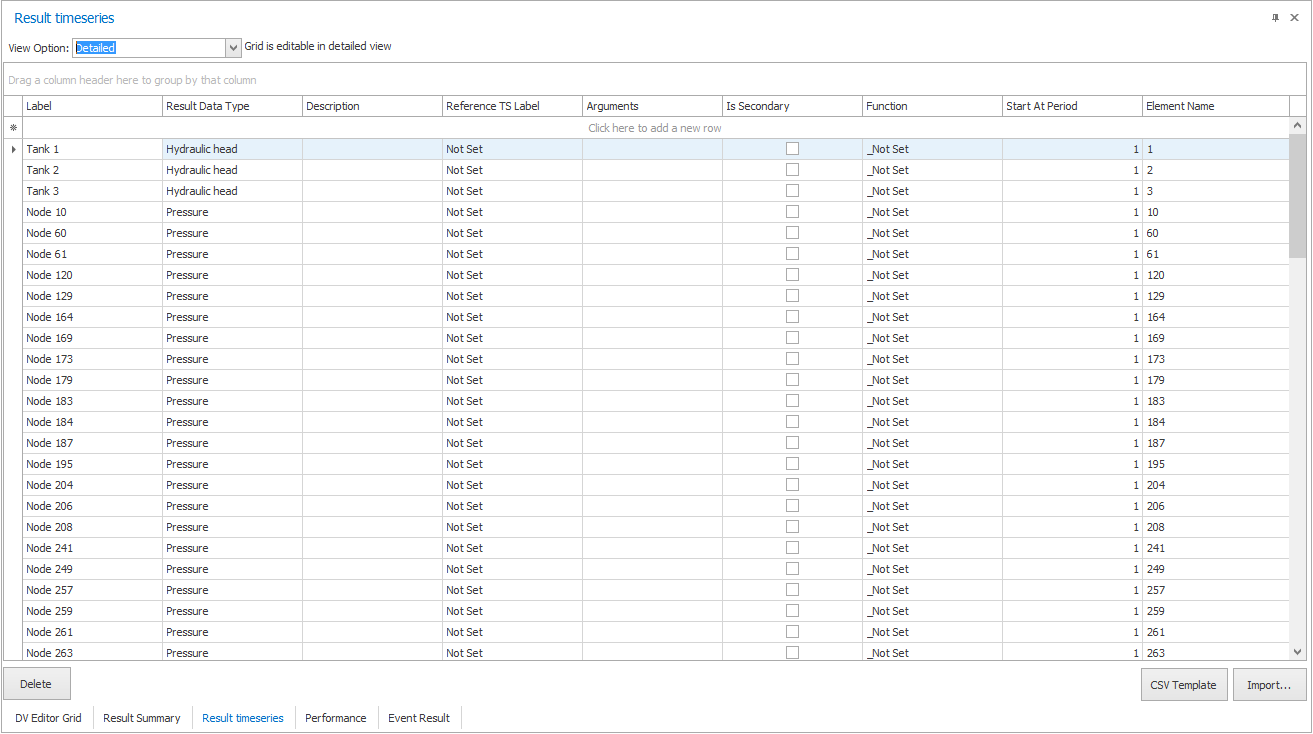
## Evaluation Configuration – DV Editor Grid

* Add a new Decision variable by clicking in the top row or through the CSV Template / Import functionality
  + Label: the abstract name for the decision variable
  + Type: the element property in the model that will be modified
  + Option: The list of possible values that the element property will take
  + Min / Max Option: From the assigned ‘Option’ defines the possible values that the element property will take
  + New Val Method: Defines how the new option value will be calculated. Typically, through Lookup from Option list specified
  + Elements: the list of real element labels in the model that the abstract decision variable is responsible for controlling
  + Primary DV / Secondary DV – used to set if a decision variable should take it’s value off another decision variable. i.e. is not allowed to change independently
  + Operation: how to apply the option value – in place of or some form of operation to the current value already in the model
* Delete a decision variable using the delete button



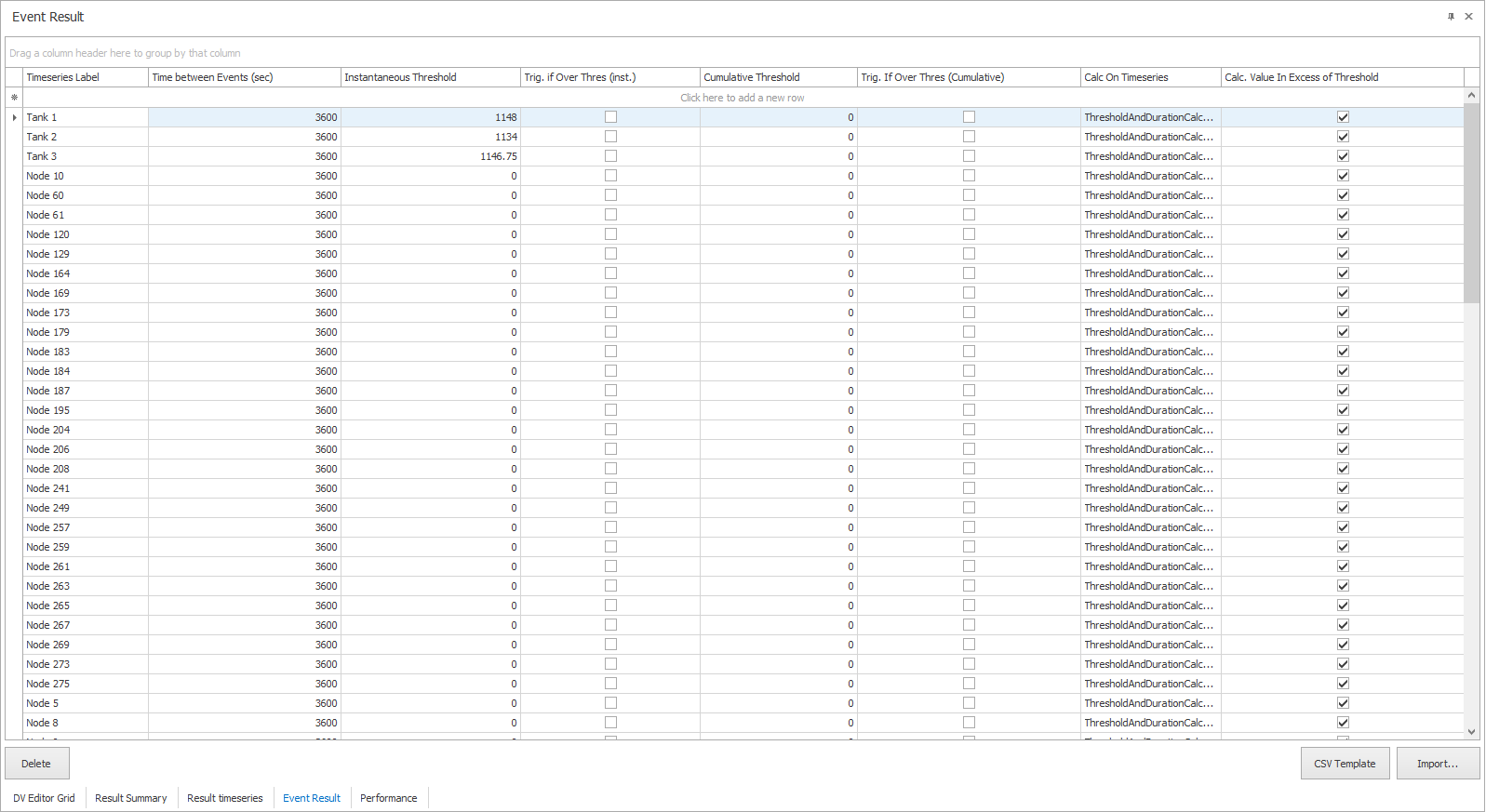
## Evaluation Configuration – Result Time Series

* Add a new time series to extract from the model simulation by clicking in the top row or through the CSV Template / Import functionality
  + Label: the abstract name for the time series that will be extracted
  + Result Data Type: the type of data that will be extracted from the model, this helps define what element the data is coming from and the property for that element
  + Start At Period: Defines the period the time series data will start
  + Element Name: the real name of the element in the model
* Delete a Result Timeseries record using the delete button



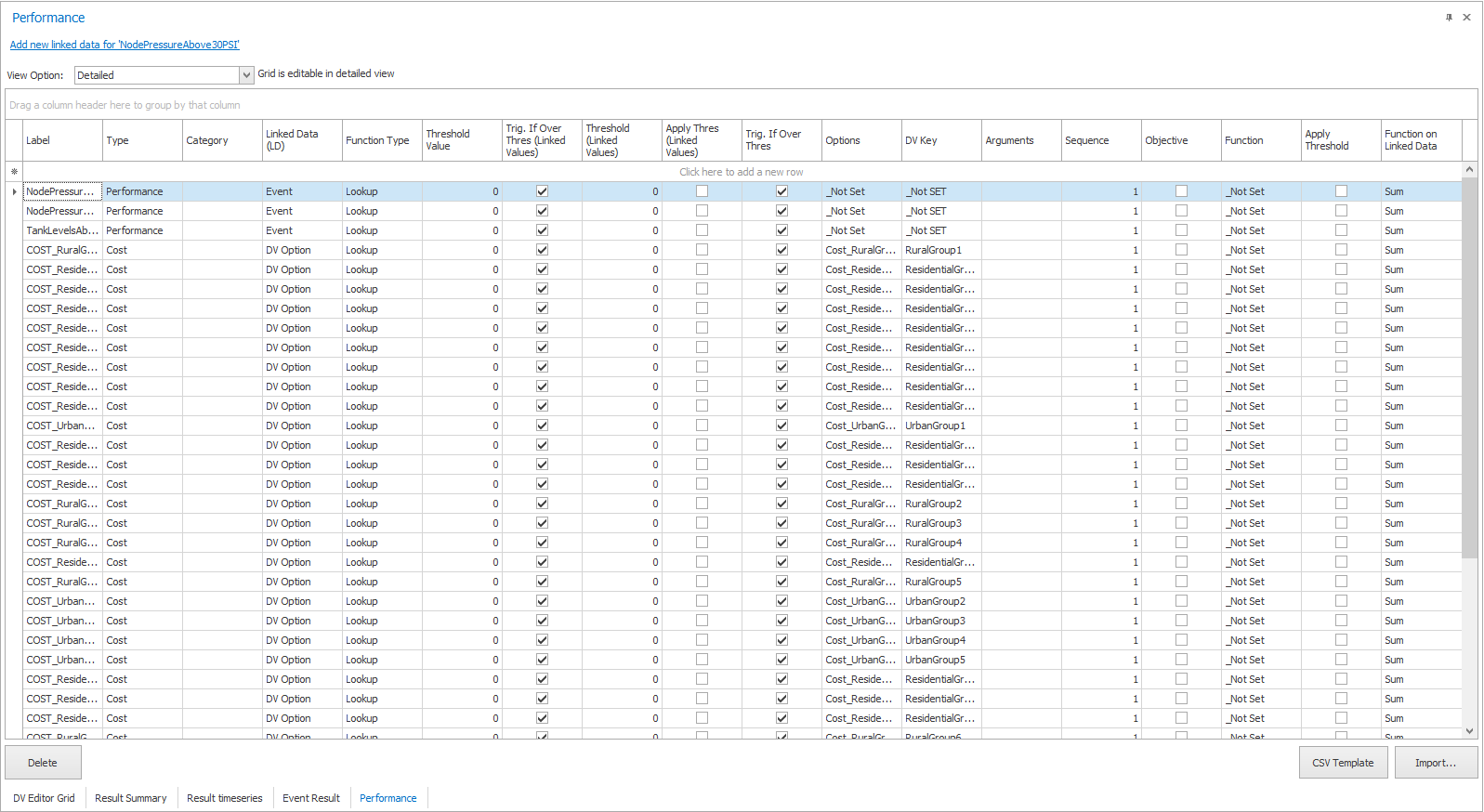
## Evaluation Configuration – Event Result

* By clicking in the top row or through the CSV Template / Import functionality, add a new event result to report an occurrence of a timeseries / another event value below or above a threshold
  + TimeSeries Label: drop down list of timeseries (and soon to be events) that can be reported on. Note that if using the CSV Import feature, currently need to enter the ResultsTS\_ID rather than the label from the database
  + Time between events: minimum number of seconds to distinguish between reporting two independent thresholds being met
  + Calc on Timeseries: currently just one default of -1
  + Instantaneous Threshold: trigger occurrence if instantaneous values of timeseries meet this value
  + Trig. If Over Thres (inst): determines whether to trigger if the instantaneous timeseries value is above or below the threshold
  + Cumulative Threshold: trigger occurrence if cumulative summation of the values in timeseries meet this value
  + Trig. If Over Thres (inst): determines whether to trigger if the cumulative summation of the values in timeseries is above or below the threshold
  + Calc Value in Excess of Threshold: determines whether the returned metric summates real value or the only the difference in value from the threshold
* Delete an Event Result record using the delete button



## Evaluation Configuration – Performance

* By clicking in the top row or through the CSV Template / Import functionality, add a new Performance record to aggregate values from Result Summary, Result Timeseries, Event Result and other Performance records for evaluating the overall performance of a scenario
  + Label: Name of the performance metric
  + Type: determines whether metric is a cost or performance
  + Function Type: Determines if data is location from a ‘lookup’ in an existing table or a ‘function’ on data in an existing table
  + Category: Currently unused but can be used to group data together for post-processing
  + Linked Data (LD) – the table containing the records that will be used for aggregating to calculate the performance metric
  + Function of Linked Data – aggregation method
  + Function: The function that will be performed if Function Type = Function
  + Objective: Checkbox to determine if this metric is the objective that will be used for optimization algorithms
  + Sequence: If some metrics require calculating first, this sequence determines the ordering of the performance calculations
  + Arguments: input arguments for the function if Function Type = Function
  + DV Key: If Linked Data = DV Option, this field is used to identify which decision variable (and it’s options) drive the performance value. This is used for cost metrics to calculate the cost of a decision variable taking on a particular option
  + Options: If Linked Data = DV Option, this field is used to identify the option list from which the performance value is set. The option value within the list is as indicated by the DV Key DV
  + Apply Threshold, Threshold Value, Trig if Over Thres: Parameters to allow filters on the aggregated performance metric
  + Apply Threshold (Linked Values), Threshold Value (Linked Values), Trig if Over Thres (Linked Values): default parameters to allow filters on the individual components comprising the aggregated performance metric
  + Add new Linked data for {Performance Label}: Identifies which records from the table Linked Data (LD) are aggregated to calculate the performance metric. Note that each individual record can have an overridden Applythreshold, Threshold and IsOver Thres. **No option to import these through CSV** – therefore suggest doing so through DB if there are a large number of entries. The ID of the linked table will need to be known



# Command Line Interface (CLI)

## Setup

The following programs/files need to be saved to the location in which Simlink will be executed in order to run simlink through the CLI. Each of programs/files is located on the CHC server at //hchfpp01/WBG/….. . Below lists each program and a recommended location for the files.

Simlink – C:\a\simlink

Modelling platform installed

HDF5 - HDF5DotNet.dll- C:\a\HDF5DotNet-Net4.0-x86

HEC-DSS - DSSUTL.EXE - C:\Program Files (x86)\HEC\HEC-DSSVue (only required if outputting the time series to DSS as specified in the config.xml)

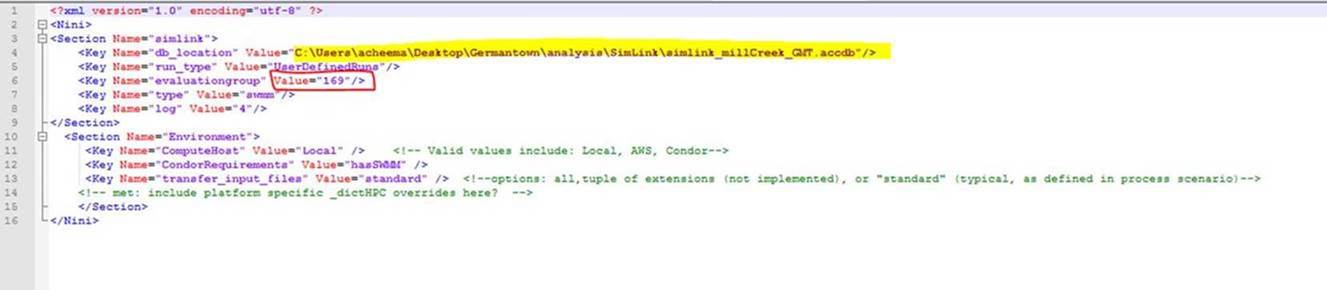
Update the system environment variable path with the folder where the exe files HDF5, DSSUTL and any modelling platform dlls reside. Navigate to the Environment Variables window (Control Panel 🡪 System 🡪 Advanced System Settings). Below is an example of system path with the relevant file locations to run simlink highlighted. Note, every time the system path is updated, the CLI must be reopened to recognize the updated path (and potentially a restart of the machine).

**Example Environment Variables system path =** C:\Program Files\Git\cmd;C:\Program Files (x86)\Microsoft SQL Server\Client SDK\ODBC\130\Tools\Binn\;C:\Program Files (x86)\Microsoft SQL Server\130\Tools\Binn\;C:\Program Files (x86)\Microsoft SQL Server\130\DTS\Binn\;C:\Program Files (x86)\Microsoft SQL Server\130\Tools\Binn\ManagementStudio\;C:\Program Files (x86)\EPANET2;C:\a\HDF5DotNet-Net4.0-x86;C:\Program Files\Microsoft SQL Server\Client SDK\ODBC\110\Tools\Binn\;C:\Program Files (x86)\Microsoft SQL Server\120\Tools\Binn\;C:\Program Files\Microsoft SQL Server\120\DTS\Binn\;~~C:\a\simlink\Engine\SimLink\_CLI\obj\Debug\~~;C:\Program Files (x86)\EPA SWMM 5.0;C:\Program Files (x86)\HEC\HEC-DSSVue

## Run

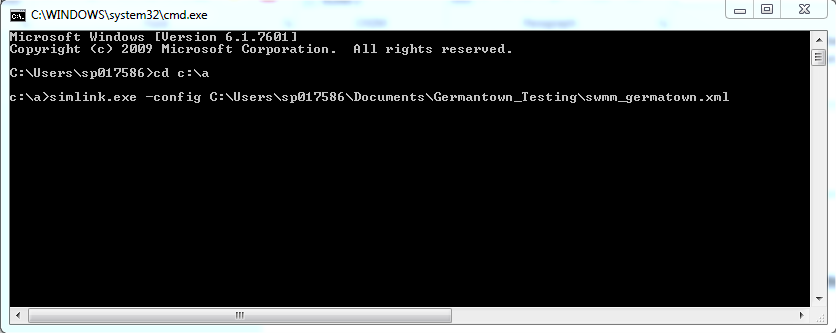
* Obtain the correct configuration file (xml) for the simlink analysis being performed (ie modeling platform, type of timeseries results extracted, etc).
* Save xml to relevant file location. The xml is typically saved in the same folder as the simlink database.
* Update the xml with the location of the simlink database (highlighted in Figure 8.1) and the evaluation group to run (boxed in red in Figure 8.1). Only scenarios in the simlink database tblScenario that have [HasBeenRun] = 0 and are in the evaluation group in the config file will be run.

**Figure 8.1** –Example Configuration File



* Navigate to the command prompt (type cmd into start menu)
* Change the directory to the location of the Simlink.exe file using **cd file location**. Example shown in Figure 8.2.
* Then type in **simlink.exe – {config full path + file name** **(of the xml config file)}**. Example shown in Figure 8.2.

**Figure 8.2** – Example of entries into CLI



## Troubleshooting

1. Error in the DBVersionValidation check with every script failing essentially indicating that there are no tables in the database when using an **MSAccess database**.
   1. Solution was to install AccessDatabaseEngine.exe on the machine
2. Error saying Exception: Could not find a part of the path to inp file.
   1. Remove any spaces within the INP file of the swmm baseline model.
   2. Run scenario from [ScenStart] = -1 to [ScenStart] = 20 to ensure all steps are included.
3. Error saying “Process Scenario Error. Last Stage 6. Error: Index was outside of the bounds of the array”
   1. Change SWMM Model version to 5.0.022
   2. Check that within cmd, you can run\_swmm5.bat that is created by Simlink in the scenario specific folder
   3. Run the baseline model within the SWMM interface to make sure it works successfully.
   4. Ensure the path of the required HDF5 dlls are located in the system path environment variables