EnterpriseArchitect Data Import Tool

Version 3.2.0 (29.2.2016)

Contents

QPR Software Plc
Huopalahdentie 24
FI-00350 Helsinki, Finland

Tel. +358 290 001 150
Fax +358 290 001 151
www.qpr.com

Customer Care +358 290 001 155
customercare@qpr.com

Domicile Helsinki
Business ID 0832693-7
VAT number FI08326937

1 Overview

EnterpriseArchitect Data Import Tool is able to

- create elements into EA/PD model and set their attributes (also relation attributes)
- update attributes of existing elements
- delete elements (when an element doesn't exist in the import data)

The tool is not able to:

- draw anything to diagrams (i.e. to instantiate elements)
- export from EA/PD model to Excel (that can be done by exporting an Analysis view from QPR Portal)
- edit EA/PD meta model

Updating is based on an identifier attribute. The identifier attribute is used to make matching between import source data and EA/PD model content, to determine whether an element exists in the EA/PD model. If the element exists, an updating is done; if it doesn't, the element is created. As a best practice, symbol is usually the best option for element identifier, but name can also be used. From the tool point of view, any attributes can be used as an identifier. The identifier values should never change, because the tool is not able to change identifiers (unless the identifier is changed).

Deleting is done when there is an element in EA/PD model which doesn't exists in the import source data. Matching is done based on the identifier attribute as when updating.

If identifier is not available in the source data or configured, updating or deleting is not possible (all data is imported as new elements).

By default column configured as attribute "symbol" acts as an identifier. It's also possible to use any other attribute as an identifier (see configuration below).

The identifier attribute must be unique and not empty both in EA/PD model data and import source data. If that is not the case, do not use that attribute as an identifier attribute.

The import source data is read from an Excel file. Import is based on a configuration document written as XML. The configuration determines e.g. what data is read form the Excel and how the data is updated to EA/PD model. The XML configuration may be stored in an EA/PD model or kept as a separate file.

To run the tool QPR EA/PD client and MS Excel must be installed in the computer.

The tool is tested in QPR 2015.1 and QPR 2014.1. The tool is probably also compatible with QPR 2012.2 and QPR 2012.1.

2 Configuring the tool

The tool can be run from

- EA/PD script file
- procedure attribute
- command line using Windows Script Host

EA/PD script file is locally stored for every workstation running the script. The advantage of a procedure element is that the script is stored as EA/PD content available to all client users. The Windows Script Host is needed when the data import is automated (i.e. set to run in Windows Task Scheduler).

To configure the tool to run from an EA/PD script file:

1. Copy file **EnterpriseArchitect Data Import Tool.qprpsc** to C:\ProgramData\QPR Software\QPR 2014\2014.1\Clients\pgscripts. (This step needs to be done only once for a workstation.)
2. Check that the following line is commented in the script (there is a ' character in the beginning of the line): **'Dim PGApplication, PGModel**.

To configure the tool to run from a procedure attribute:

1. Setup an EA/PD element type which for example can serve as a system element for data import. The element type must have following two custom attributes:
    a. Procedure type of attribute containing the data import script code (copied from **EnterpriseArchitect Data Import Tool.qprpsc**). The script code is recommended to be stored for the custom attribute type itself, so that the script is available by default for all data import elements.
    b. Memo type of attribute, which is for storing the XML configuration (usually it is set for each element separately). Default name of the attribute is **Configuration**. The name of the attribute need to be saved in the beginning of the script (see chapter 8. Tool parameters).
2. Check that the following line is commented in the script (there is no ' character in the beginning of the line): **Dim PGApplication, PGModel**

To configure the tool to run automatically from the Windows Script Host:

1. Copy **EnterpriseArchitectDataImportTool.vbs** to the computer where the script will be scheduled to run. (The extension **.vbs** is mandatory for the script file.)
2. Check that the following line is not commented in the script (there is no ' character in the beginning of the line): **Dim PGApplication, PGModel**
3. Test the script using command **cscript [script location] [XML configuration location]**, e.g. **cscript C:\EnterpriseArchitectDataImportTool.vbs C:\ImportConfiguration.xml**. Use logging, because all errors appear in the log.
4. Create a task in the Windows Task Scheduler. Use the same command for the task.

3 Running the tool

Before running the tool, the tool needs to be configured according to chapter 0. The tool is tested in QPR 2015.1 and QPR 2014.1. The tool is probably also compatible with QPR 2012.2 and QPR 2012.1.

Configuring the tool.

Follow these steps to run the tool from an EA/PD script file:

1. Open a model in **EA/PD Client.**
2. Select **Home** > **Run Script** > **EnterpriseArchitect Data Import Tool.**
3. If an element is selected containing import XML configuration, that configuration is used. Othwerwise an XML configuration file is asked from user.
4. If the XML configuration doesn't reference to an Excel file, also an Excel file is asked from user.
5. Wait until the import is ready and a notification message appears. Note that the model is not saved, so you can reverse the changes by closing the model without saving it.

Follow these steps to run the tool from a procedure attribute:

1. Select an element configured for data import.
2. Click the secondary mouse button (to open the popup menu), and select **Execute Procedure** > **EnterpriseArchitect Data Import Tool**.
3. If the element contains the import XML configuration, that configuration is used. Otherwise an XML configuration file is asked from user.
4. If the XML configuration doesn't reference to an Excel file, also an Excel file is asked from user.
5. Wait until the import is ready and a notification message appears. Note that the model is not saved, so you can reverse the changes by closing the model without saving it.

4 Excel format for source data

Excel files read by the tool have the following structure and limitations:

- Data is processed in the order they appear in the XML configuration. This should be taken into account if there are relations between the imported elements (e.g. an element must be imported before a relation to the element can be set).
- It's possible to read data from multiple Excel files based on a single XML configuration.
- Each sheet may contain only one type of EA/PD elements. (Examples of element types are Activity, Subprocess, Organization item type, Information item type.)
- A sheet may contain a header row which can be used to identify columns in the XML configuration (instead of column numbers).
- The tool stops reading a sheet when a row where all the cells are empty is encountered.
- For cardinality N attributes, each attribute value can be defined in a separate row (see example data). All attributes of a single element must be in consecutive rows. Other cells of the additional rows should be left empty.
- For relation elements, any attribute of the target element can be used to identify the element (the attribute should be unique, of course).
- Merged cells are not supported. All merged cells must be removed before importing.

5 XML configuration

The XML configuration maps the source data in Excel to the EA/PD meta model by defining what data is read and where it is saved in an EA/PD model. A single XML configuration can read data from several Excel files and several worksheets of a single Excel file. The XML configuration have following hierarchical XML tag structure:

```
dataimport (1)
  importfile (1..n)
    worksheet (1..n)
      column (1..n)
```

The indentation represents parent-child tag relationships. Possible number of tags is presented in parenthesis: there is always one **dataimport** element, and other elements can be many.

Following tables lists available tag attributes. Mandatory attributes are mentioned; others may be left out.

Tag: **dataimport**

Tag attribute	Description
logfile	Import log file location. If no log file is specified, logging is disabled. Logging is appended to an existing file so no log is lost by running the tool. Log columns are tab separated and they can be copied to Excel. The logging stores all element creation and attribute save operations to EA/PD model.
addtimestamptologfile	Alternatives **true** or **false**. If true, adds a timestamp to the name of the log file. As a result, every run will have its own log file.
servername	QPR server dns name. Used when the import is run from the Windows Script Host.
serverport	QPR server port. Used when the import is run from the Windows Script Host.
username	QPR user name. Used when the import is run from the Windows Script Host.
password	QPR user password. Used when the import is run from the Windows Script Host.
modelname	QPR EA/PD model name. Used when the import is run from the Windows Script Host. When the script is run from the designer, the import is done to the opened model, and this setting has no effect.
modellinglanguage	Language code of the modelling language which element and attribute names in the xml configuration are referencing to.

Tag: **importfile**	
Tag attribute	**Description**
location	Source Excel data location in the disk (full file path). If left out when running from the designer, the location is asked from user. For running from the Windows Script Host, this attribute is required.
folder	When "location" is not defined and thus the source Excel file needs to be selected manually, file browser window is opened in this default path. This attribute is only used when running from the designer.
filetype (default "excel")	Type of source data file. Only **excel** is currently supported.

Tag: **worksheet**	
Tag attribute	**Description**
sheetname	Name of the Excel worksheet where data is read from. Sheets may also be referenced using order numbers with attribute "sheetnumber".
sheetnumber	Order number of Excel worksheet starting from left (the leftmost sheet is **1**). Sheets may also be referenced using sheet names with attribute "sheetname".
elementtype (mandatory)	Imported EA/PD element type name. One worksheet may thus contain only one type of elements.
deletemode	Either **true** or **false** indicating whether not found elements should be deleted in the

(default "false")	EA/PD model. Deleting may be disabled with this setting, because it may cause inadvertent loss of data.
headerrow (default "1")	Header row number in the Excel sheet (topmost row is **1**). If columns are referenced using header names (i.e. "columnheader" attribute in "column" tag), this is the row to read the column headers.
firstdatarow (default "2")	Row number for the first data row (topmost row is **1**). The import starts from this row and reads data downwards until an empty row is found.

Tag: **column**	
Tag attribute	**Description**
attribute (mandatory)	Name of the EA/PD attribute to which data is imported from the corresponding column. Valid attribute names are the ones used by QPR API (documentation is available here: http://kb.qpr.com/qpr2014-1/index.html?setproperty3.htm).
columnheader	Excel column header name. This information is used to determine from which Excel column the information is read. This is alternative to attribute "columnnumber".
columnnumber	Excel column order number (the leftmost is **1**). This information is used to determine Excel column. This is alternative to attribute "columnheader".
type (default "text")	Identifies EA/PD attribute type. Possible options are - **relation** (for relation attributes) - **text** (for other type of attributes) If value is **relation**, also parameters "relationto" and "identifierattribute" need to be defined.
updatemode (default "synchronize")	Determines how attributes are updated from Excel to EA/PD. For cardinality 1 attributes options are: - **synchronize**: Updates all values from Excel to EA/PD including empty values. - **nonemptyonly**: Updates only non-empty values to EA/PD (i.e. empty values in Excel are not updated to EA/PD). - **addonly**: Only updates values that are previously empty in EA/PD (i.e. value is not updated if EA/PD already contains a value) For cardinality N attributes options are: - **synchronize**: Set all relations to correspond to the Excel data (i.e. relations not existing in the Excel are removed). - **addonly**: Relations from the Excel data are only added, and no existing relation in EA/PD are removed. The table below illustrates the differences between these options.
relationto (no default value)	Relation target element type name (i.e. type of elements which the relation is pointing to). This must correspond to EA/PD meta model. This is only defined for columns having type="relation".
cardinality (default "1")	Either **1** or **N** corresponding to EA/PD meta model.
identifierattribute	Relation target element identifier attribute name. E.g. when "symbol" is typed, write

(no default value)	elements symbols in this column. This is only defined for columns having type=”relation”.
isidentifier (default ”false”)	**true** defines that this attribute is used as an identifier for the updating logic. By default column referencing to “symbol” attribute is used as an identifier, even if it’s not defined to be an identifier by this attribute. If no identifier exists (either defined explicitly by this attribute, or existence of symbol attribute), no updating or deleting is done (only creating).
defaultvalue (no default value)	If Excel contains an empty value (empty cell), this defined default value is updated instead of the empty value. Updatemode attribute works as if the default value had been stored in Excel.

The following table illustrates the differences between **updatemodes** depending whether the source or target attributes have a value.

Attribute value in		**updatemode is**		
Excel (source)	**EA/PD (dest.)**	**synchronize**	**nonemptyonly**	**addonly**
Non empty	Empty	updated	updated	updated
Non empty	Non empty	updated	updated	not updated
Empty	Non empty	updated	not updated	not updated

6 Example configuration

Below is an example XML configuration file.

```
<?xml version="1.0" encoding="utf-8"?>
<dataimport xmlns="http://www.qpr.com/QPREnterpriseArchitect/DataImport" logfile="C:\DataImportLog.txt" servername="localhost" serverport="4251"
username="qpr" password="demo" modelname="\\Dentorex - Quality Management System">
  <importfile location="C:\Example data.xlsx" filetype="excel">
    <worksheet sheetname="Controls" elementtype="Control" deletemode="false" headerrow="1" firstdatarow="2">
      <column attribute="name" columnheader="name" />
      <column attribute="description" columnnumber="2" />
      <column attribute="symbol" columnheader="symbol" />
      <column attribute="Control Test Outcome" columnnumber="4" />
      <column attribute="Risks for Control" columnheader="Related risks" type="relation" relationto="Risk/Riskit" cardinality="N"
identifierattribute="symbol" updatemode="synchronize" />
    </worksheet>
    <worksheet sheetnumber="2" elementtype="Risk/Riskit">
      <column attribute="name" columnnumber="1" />
      <column attribute="symbol" isidentifier="true" columnnumber="3" />
      <column attribute="description" updatemode="nonemptyonly" columnnumber="2" />
      <column attribute="Risk sub category" columnheader="Sub category" />
    </worksheet>
  </importfile>
</dataimport>
```

Example of the simplest possible configuration:

```
<?xml version="1.0" encoding="utf-8"?>
<dataimport xmlns="http://www.qpr.com/QPREnterpriseArchitect/DataImport">
  <importfile location="C:/Example data.xlsx" filetype="Excel">
    <worksheet sheetname="Sheet1" elementtype="Control">
      <column attribute="name" columnnumber="1" />
    </worksheet>
  </importfile>
```

</dataimport>

7 Debugging in Excel development tools

The script code can be developed and debugged in Excel in Visual Basic editor. The following changes are needed so that the script runs in Excel:

- set **EXCEL_DEBUGGING=True**
- remove comment in line **Dim Wscript**
- Add comment in line **'startProgram()**
- remove comment in line **Dim PGApplication, PGModel**
- set variable **Const TEST_CONFIGURATION_FILE**

8 Tool parameters

The tool contains the following parameters, which are defined in the beginning of the script file. These parameters don't usually need to be changed.

Parameter	Description
XML_CONFIGURATION_ATTRIBUTE	EA/PD element attribute which contains import XML configuration. Defined using QRP API name.
DEFAULT_CONFIGURATION_FILE_PATH	When XML configuration file location needs to be selected manually when running from the designer, file browser window is opened in the default path. This parameter is only used when **folder** attribute of the **importfile** tag is not defined.
DEFAULT_EXCEL_FILE_PATH	When source Excel file path needs to be selected manually, file browser window is opened in the default path.
DEFAULT_IDENTIFIER	Attribute which is used as an identifier (in the update logic) if no other identifier is explicitly defined in the XML configuration. Defined using QRP API name.
EXCEL_DEBUGGING	**True** or **False**. Set to True when the script is run from Excel (usually for debugging purposes).
TEST_CONFIGURATION_FILE	Path containing test configuration file which can be used to make debugging in Excel easier.

9 Future improvements

- Easy importing hierarchical data (this is a concept, no spec is ready)