Design Specifications of the WaM-DaM Wizard

Adel M. Abdallah and David E. Rosenberg

September, 2016

1. Introduction

This document describes the design specifications of the “WaMDaM Wizard” Graphical User Interface. WaMDaM Wizard helps and guide users to load data from spreadsheet templates to a WaMDaM SQLite database instance according to the Water Management Data Model (WaMDaM) specifications (**Figure 1**). The Wizard will be designed as a desktop application that works across the Windows, Mac, or Linux platforms using the wxPython library based on Python 2.7.

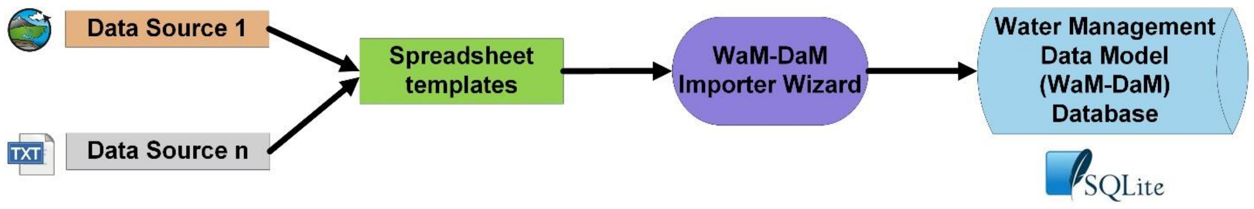


Figure 1: Conceptual diagram showing the role of the WaMDaM Wizard to help users load data from spreadsheet templates into the WaMDaM database.

WaMDaM is a relational schema that is designed to organize and compare systems water management data from multiple disparate datasets and study locations (**Figure 2**). Systems water management data include binary, numeric and text parameters, time series, and multi-column arrays that describe attributes of networks comprised of node and link water systems components. The design of WaMDaM uses controlled vocabularies to consistently relate terms across data sources, system components, and their attributes. WaMDaM design has been tested and a Data Definition Language (DDL) scripts to create blank schemas are available for SQLite, Microsoft SQL Server, MySQL, and PostgreSQL. However, the Wizard will be designed to work only with SQLite because SQLite is free, open-source, and does not need a server. The populated WaMDaM SQLite instance file could be shared and published online like at https://www.hydroshare.org/. WaMDaM, the Wizard, and all their documentations are open-source and distributed under a BSD 3-Clause license. The Wizard and its documentation files will be published on GitHub. This document will not describe the data entities, their relationships, and why they are designed in this particular way. For more info about WaMDaM design, check out the GitHub repository @ https://github.com/amabdallah/WaMDaM

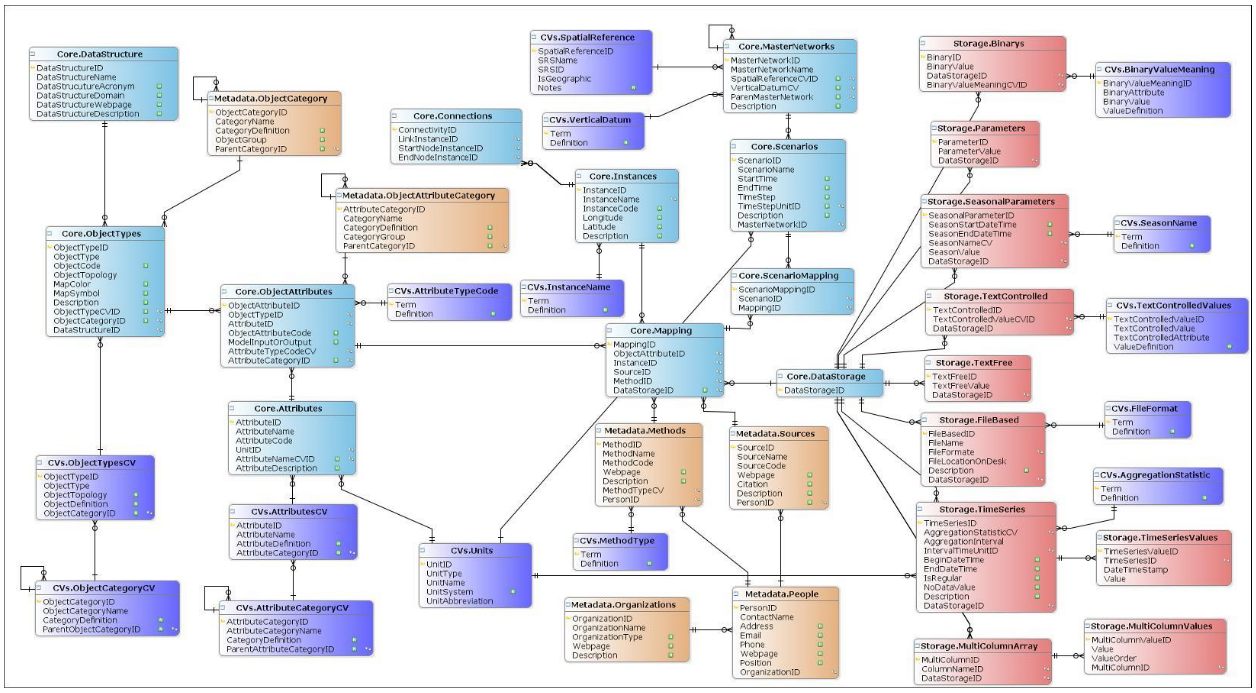


Figure 2: WaMDaM Schema. Light blue represents the core entities of data structures and networks that are always used in loading data into WaMDaM, light orange represents metadata entities, purple represents controlled vocabulary entities, and light red represents entities that store data values

The general concept behind WaMDaM Wizard is that it accepts input data from a specific Microsoft Excel table format, which can be loaded into WaMDaM without violating any WaMDaM self-business rules or specified software constraints. The Wizard front-end that provides visual interaction and controls has been designed using wxFormBuilder software https://sourceforge.net/projects/wxformbuilder. The Wizard design was initially envisioned in mockup windows that show the steps that users will follow while using the Wizard to load their data into a SQLite database. The mockups are accessible online at https://app.moqups.com/amabdallah/6pbEfKbbDx/view/page/a0bbb4b4a. However the wxFormBuilder file is the final front-end design to consider. Here, users will take these following steps in using the WaMDaM Wizard:

1. Download the WaMDaM Wizard installer to a local machine. The installer will come with a blank WaMDaM SQLite database.
2. Populate the blank spreadsheet templates provided on GitHub with input data from one or more datasets.
3. Launch the Wizard and connect to the SQLite WaMDaM database instance.
4. Follow the Wizard GUI to load data values from each spreadsheet to the database in the right sequential order in three major steps: i) import Data Structures, ii) import Networks, and iii) import Data Values. The provided WaMDaM instance will be already populated with example controlled vocabularies and metadata but users can add new controlled vocabularies and metadata within the user interface as they follow the three steps above as I explain later.

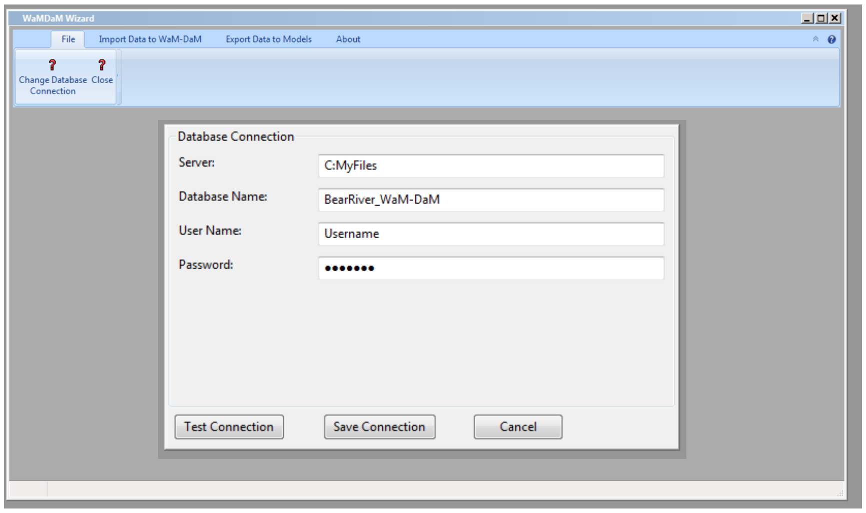
Successfully loading the data into the database requires the user to comply with the database and software business rules as elaborated later. The goal and benefit of the Wizard to users is three-fold: first to guide users to load their data into WaM-DaM without the need to use foreign keys or understand how the tables in WaM-DaM are connected. Second, to relate different terms with controlled vocabulary and provide contextual metadata with minimal effort and third, to perform automated validation of data entries to comply with WaMDaM specifications.

This document describes the steps, in order, that the users will follow from installing the Wizard up to successful message of loading the last data value. After that, the document describes technical design details. The following section describes the installation and connection to the database. Sections 3 describes the three major steps to import data from spreadsheets to the database. Section 4 describes how to import additional metadata and controlled vocabularies to the WaMDaM database. Section 5 elaborates on the Wizard part called “About WaMDaM”. Section 6 discusses data validation, integrity checks, and transaction management. Section 7 describes the architecture of the Wizard software.

2. Install WaMDaM Wizard and connect to SQLite database

The Wizard will be available to download on GitHub as a standalone installer software that works on the Windows, Mac, and Linux operating systems. The software installation process will install all of the necessary components and files including SQLite for the WaMDaM Wizard application to work. Once the user installs the Wizard, they can launch it by clicking on a shortcut icon on their desktop or through Start Menu.

Upon launching the Wizard, a “Connect to SQLite Database” window will pop-up with box entries that require all the login authentication information: Server Address, Database Name, Server User ID, and Server Password (**Figure 3**). Users can click “Test the Connection” to verify that it works. If the test is successful, the “Save Connection” button will be activated and users may save the connection authentication info for future login. Users could choose to cancel the connection and exit. Upon a successful login, the Wizard homepage will pop-up. Users can change the database connection to another database instance by clicking on File tab in the homepage ribbon Menu and then they click on “Change Database Connection” in the drop down menu. Next, users will be able to import data from spreadsheet templates into a blank WaMDaM SQLite database as described in Section 3.



Database Connection

Server: C:MyFiles

Database Name: BearRiver_WaM-DaM

User Name: Username

Password: •••••••

Figure 3: Homepage of the WaMDaM Wizard with the database configuration window. The button "Save Connection" is inactive until the connection is tested to be valid. Then the user can save their valid connection.

3. Import Data to WaMDaM from spreadsheet templates to the WaMDaM database

Once the user has successfully connected to the database, they can click at the Tab "Import Data To WaMDaM" which reveals the ribbon bar buttons with the three major steps and their sub-steps to import data to WaMDaM (**Figure 4**). There is another useful functionality in the Wizard to "Export Data to Models" which will be added in a future update. The three major steps to import data from spreadsheets to WaMDaM are: i) import Data Structures which includes Object Types and their Attributes, ii) import Networks which includes scenarios and their nodes and links, and iii) import Data Values which include eight different data types. The steps and their sub-steps must be followed in order except the sub-steps of the third step to import Data Values which could be followed in any order (**Figure 5).**

The spreadsheet workbooks must have the extension of (.xlsx) for Microsoft Office 2007 and newer versions. You can find the workbooks at GitHub @ xxxxxx. Spreadsheets names and column headers are locked to prevent the user from changing them. Any changes in the names will prompt an error message in the Wizard that warns the user for a changed name. The purpose of keeping fixed names will allow the Wizard to automatically identify each sheet and its columns in a workbook without the need for users to do so manually. This automation will save the user the effort to navigate to each spreadsheet inside the workbook and reduce the chance of errors. Users need to only navigate once to the workbook that contains the Structures workbook and then the Wizard automatically navigates to Structures, ObjectTypes, Attributes, ObjectAttributes spreadsheet and their columns.

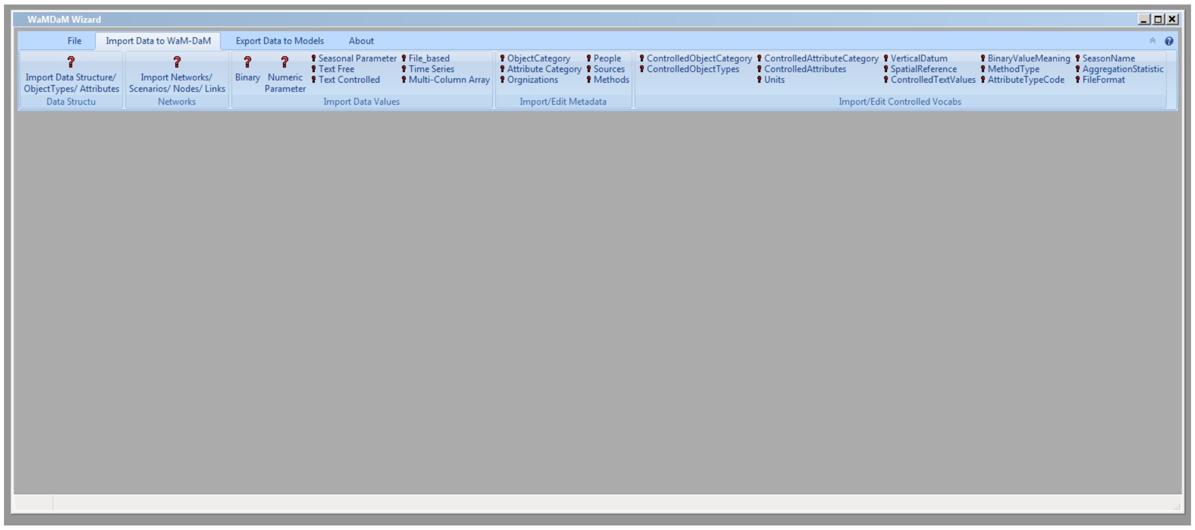


Figure 4: Snapshot of the Wizard homepage with the three major steps and their sub-steps to "Import Data To WaMDaM". I need to Figure out how to change the last two tabs to become a dropdown menu to reduce the confusion to users.

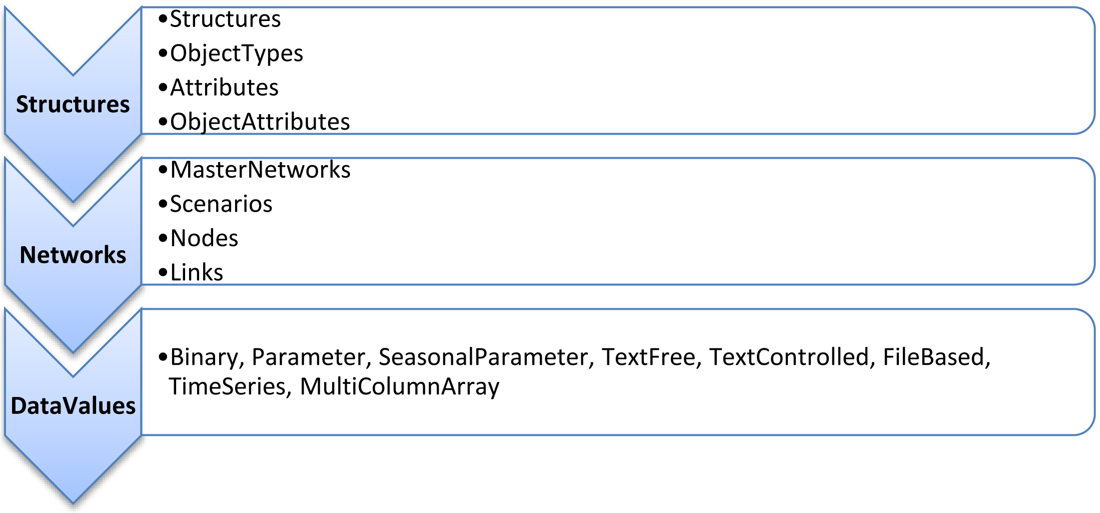
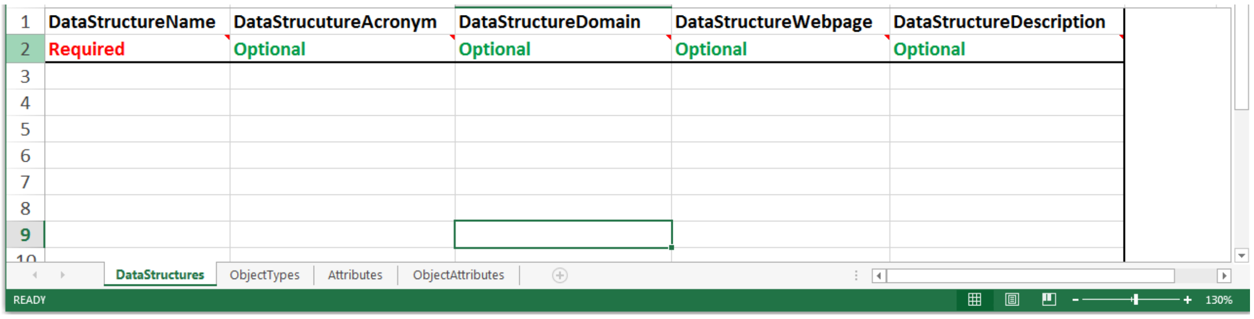


Figure 5: WaMDaM Wizard workflow steps that users take to load data from spreadsheet templates into the WaMDaM database



1	DataStructureName	DataStrucutureAcronym	DataStructureDomain	DataStructureWebpage	DataStructureDescription
2	Required	Optional	Optional	Optional	Optional
3					
4					
5					
6					
7					
8					
9					

Figure 6: Example format of the "Structure" workbook template

3.1 Import Data Structures

Defining a new Data Structure in WaMDaM consists of four steps. The user has to follow all of them to complete defining a new Data Structure (**Figure 7**). If the user stops before completing the four steps, the Wizard aborts defining the new Data Structure. The user could later edit an existing Data Structure by adding additional Object Types or Attributes.

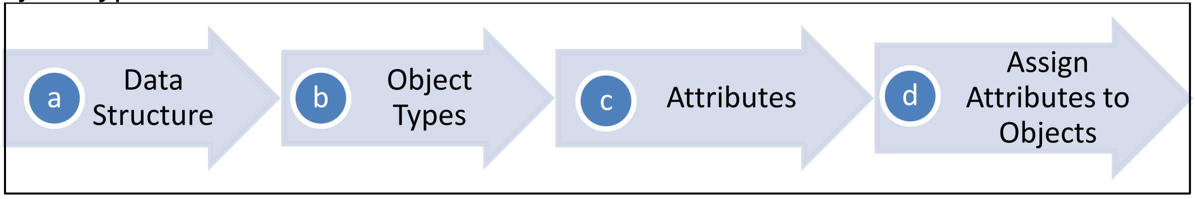
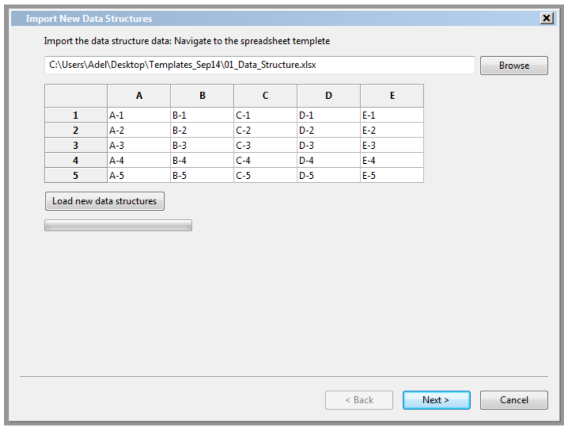


Figure 7: The four consecutive steps to import a data structure in WaMDaM

When a user clicks on “Data Structure” in the homepage ribbon menu under “Import Data To WaMDaM” in **Figure 4**, a wizard window pops up (**Figure 8**). The user can navigate to the spreadsheet workbook called “Structures” that contains all the Data Structure spreadsheets. Users can define many Data Structures at the same time. The progress bar at the bottom of the wizard window will help users visually see the percentage of their progress in following the four sub-steps to define a new Data Structure. The term “wizard window” here is used in wxPython and refers to a window with a sequence of more than two steps or pages.

Next, the user adds Object Types to a specific Data Structure they select (**Figure 9**). Selecting a Data Structure copies its primary key as foreign key to all the Object Types. When the user clicks on “Add New Object Types”, they will import all the Object Types at the same time. WaMDaM at this point is not designed to support reusing Object types from other data structures that might exist in the database. When the user clicks on the button “Load Objects” to the database, the Wizard automatically checks if the Object Names match any of the existing Controlled Vocabulary (CV) and prompts the user to choose if they want to match their term with an existing term. If there is no match, the Wizard suggests the closest terms based on the percentage of match. Otherwise, the Wizard prompts the user to either add the term as a new controlled term or to skip this step and move on without registering their term with any of the controlled vocabulary. Registering the new term against a CV copies the primary key of that CV to the ObjectType table as a foreign key. Registering CVs for other terms in WaMDaM will follow the same procedure like for Attribute names and units. Similarly registering metadata like methods, sources, Object Category follows the same procedure to register CVs above. Next, the user adds the Data Structure attributes that can be shared across all the defined Object Types (**Figure 10**). Finally, the user assigns attributes to each object (**Figure 11**). The Wizard uses the provided ObjectType and the Attribute Names in the ObjectAttribute spreadsheet to look up the primary keys in both tables and use them as foreign keys in the ObjectAttributes Table in WaMDaM. Next, the second major step to import data into WaMDaM uses the defined Data Structure here to create Networks.

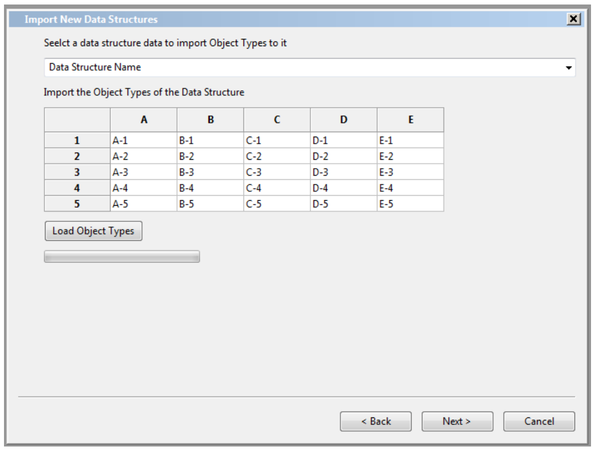


Import the data structure data: Navigate to the spreadsheet templete

C:\Users\Adel\Desktop\Templates_Sep14\01_Data_Structure.xlsx

	A	B	C	D	E
1	A-1	B-1	C-1	D-1	E-1
2	A-2	B-2	C-2	D-2	E-2
3	A-3	B-3	C-3	D-3	E-3
4	A-4	B-4	C-4	D-4	E-4
5	A-5	B-5	C-5	D-5	E-5

Figure 8: Wizard window page to import input data from the Data Structure workbook



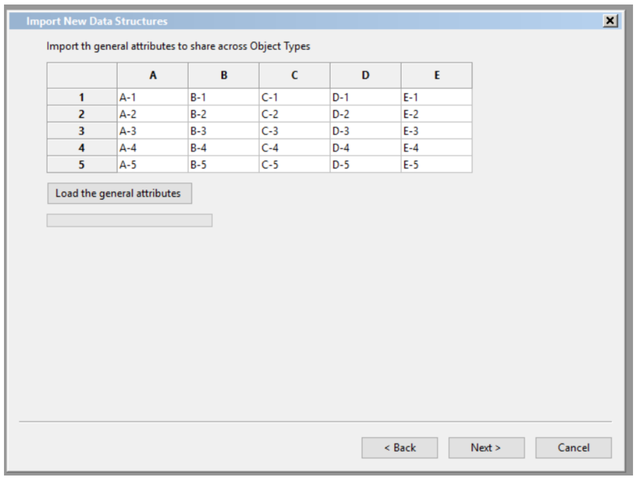
Seelct a data structure data to import Object Types to it

Data Structure Name

Import the Object Types of the Data Structure

	A	B	C	D	E
1	A-1	B-1	C-1	D-1	E-1
2	A-2	B-2	C-2	D-2	E-2
3	A-3	B-3	C-3	D-3	E-3
4	A-4	B-4	C-4	D-4	E-4
5	A-5	B-5	C-5	D-5	E-5

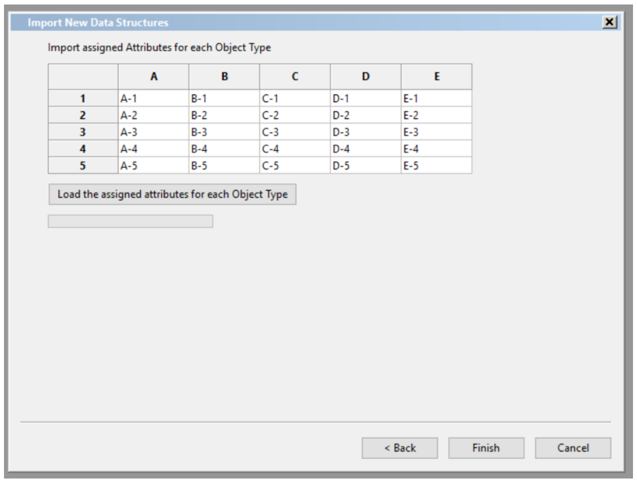
Figure 9: Wizard window page to import object types from the Data Structure workbook



Import th general attributes to share across Object Types

	A	B	C	D	E
1	A-1	B-1	C-1	D-1	E-1
2	A-2	B-2	C-2	D-2	E-2
3	A-3	B-3	C-3	D-3	E-3
4	A-4	B-4	C-4	D-4	E-4
5	A-5	B-5	C-5	D-5	E-5

Figure 10: Wizard window page to import Attributes from the Data Structure workbook


Import New Data Structures

Import assigned Attributes for each Object Type

	A	B	C	D	E
1	A-1	B-1	C-1	D-1	E-1
2	A-2	B-2	C-2	D-2	E-2
3	A-3	B-3	C-3	D-3	E-3
4	A-4	B-4	C-4	D-4	E-4
5	A-5	B-5	C-5	D-5	E-5

Figure 11: Wizard window page to assign attributes to each object

3.2 Import Networks from the spreadsheet to the WaMDaM database

There are four sub-steps to define a Network (**Figure 12**). Each action imports data from a separate spreadsheet as explained below. The user has to complete at least all the first three steps to define a new network. If the user stops before finishing the three steps, then the Wizard aborts this process to import the new network. The user can come back and add one or more additional scenarios to the network plus add more nodes and links. The network in WaMDaM belongs to a Data Structure concept, which defines data templates for the network elements being nodes and links.

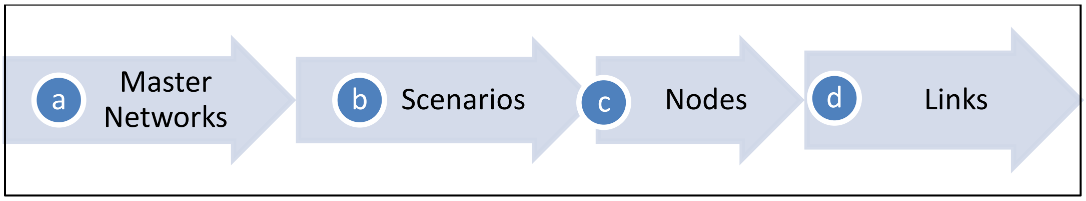
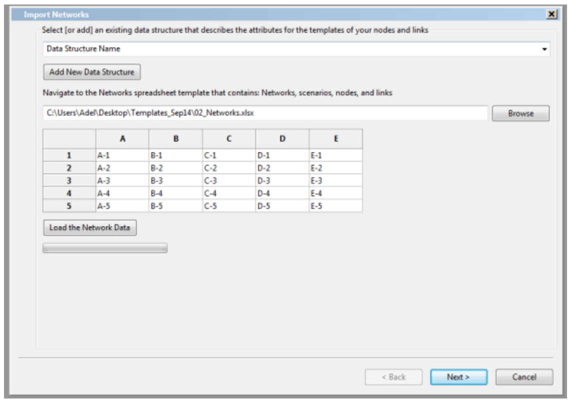


Figure 12: The four consecutive steps to import a Network in WaMDaM

When a user clicks on “Import Networks” in the homepage ribbon menu under “Import Data To WaMDaM” in **Figure 4**, a window wizard pops up with the four actions (**Figure 13**). The user has to select a Data Structure that the new Network belongs to. If there is only one Data Structure loaded in the database, the Wizard automatically selects it. If the Data Structure does not exist, the button “Add New Data Structure” takes the user to the previous section 3.1 to add a new Data Structure. The window in **Figure 13** remains open and awaits the user to add the Data Structure. Once the user has finished the last sub-step to the Data Structure, **Figure 13** refreshes to allow the user to select the newly added Data Structure. Then the user clicks “Browse” to locate the workbook template that has the Networks data. [If the Master Network already existed in WaMDaM and the user wants to add additional scenarios, nodes, or links to it, then the user can skip this action and clicks “Next”. Then the user can click “Next” in the Wizard which will show up as in **Figure 14** to define Scenarios. The user has to select a specific network to import the scenario to it. In the next two actions, the user selects the scenario and adds to it the nodes and links (**Figure 15** and **Figure 16**). Next in this section, I elaborate on the software business rules and underling coding that is required to add nodes and links for scenarios and networks.



Select [or add] an existing data structure that describes the attributes for the templates of your nodes and links

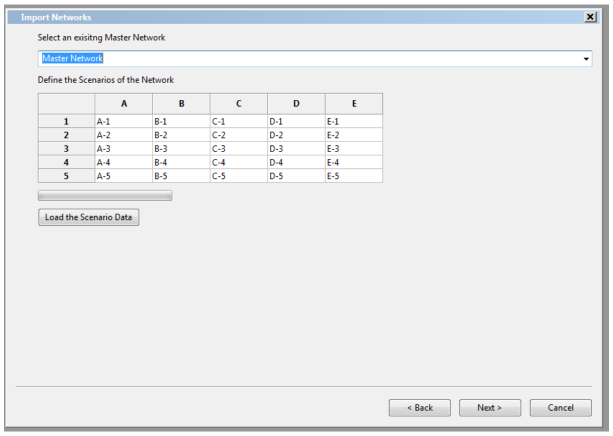
Data Structure Name

Navigate to the Networks spreadsheet template that contains: Networks, scenarios, nodes, and links

C:\Users\Adel\Desktop\Templates_Sep14\02_Networks.xlsx

	A	B	C	D	E
1	A-1	B-1	C-1	D-1	E-1
2	A-2	B-2	C-2	D-2	E-2
3	A-3	B-3	C-3	D-3	E-3
4	A-4	B-4	C-4	D-4	E-4
5	A-5	B-5	C-5	D-5	E-5

Figure 13: Wizard window page to define a new network



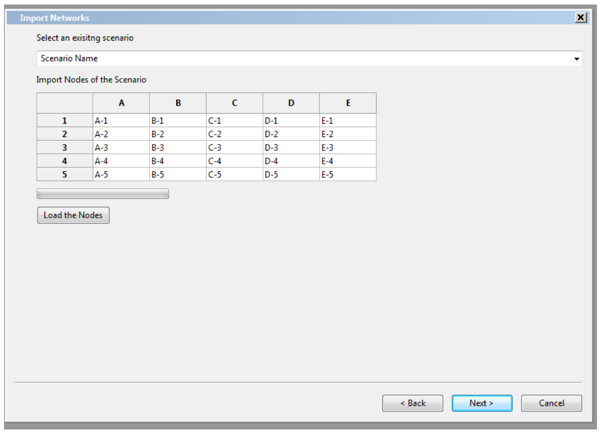
Select an exisitng Master Network

Master Network

Define the Scenarios of the Network

	A	B	C	D	E
1	A-1	B-1	C-1	D-1	E-1
2	A-2	B-2	C-2	D-2	E-2
3	A-3	B-3	C-3	D-3	E-3
4	A-4	B-4	C-4	D-4	E-4
5	A-5	B-5	C-5	D-5	E-5

Figure 14: Wizard window page to define a new scenario



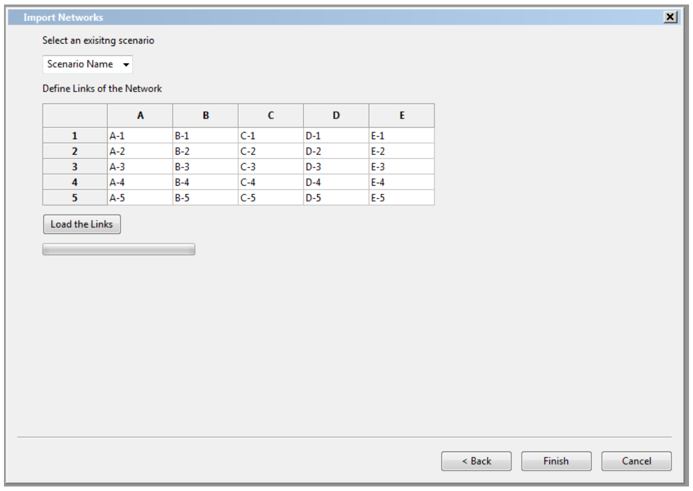
Select an exisitng scenario

Scenario Name

Import Nodes of the Scenario

	A	B	C	D	E
1	A-1	B-1	C-1	D-1	E-1
2	A-2	B-2	C-2	D-2	E-2
3	A-3	B-3	C-3	D-3	E-3
4	A-4	B-4	C-4	D-4	E-4
5	A-5	B-5	C-5	D-5	E-5

Figure 15: Wizard window page to define the scenario nodes


Import Networks

Select an exisitng scenario

Scenario Name

Define Links of the Network

	A	B	C	D	E
1	A-1	B-1	C-1	D-1	E-1
2	A-2	B-2	C-2	D-2	E-2
3	A-3	B-3	C-3	D-3	E-3
4	A-4	B-4	C-4	D-4	E-4
5	A-5	B-5	C-5	D-5	E-5

Figure 16: Wizard window page to define the scenario links

The Wizard has two tricky technical data loading steps which involve looking up and mapping foreign keys. The center of these steps that maps tables together in WaMDaM is a table called “Mapping”. This table is invisible to users and it is populated separately in two times. The first time is to load instances and related them with ObjectTypes, instances, networks, scenario, and metadata. The user can define a new network which comprise node and links instances but without populating data values for the attributes of instances. The second time is to connect an attribute of an instance within a scenario with a data value. I explain the first one now and I elaborate on the second in Section 3.3 to load Data Values. Both of them are software business rules that the Wizard has to implement.

Although the Instances Table appears to be independent in WaMDaM, there is a need for a software business rule to populate this table with data but at the same time connect it with other tables that relate it to the Object Type, Attribute, a Data Structure, Scenario, and a Network. Therefore loading new instances requires populating the Mapping Table as well to relate all the mentioned entities together. The Mapping Table has several foreign keys that are needed to be looked up based on provided metadata in the Nodes and Links spreadsheets along the selections in the Wizard as in **Figure 14**, **Figure 15**, and **Figure 16**.

Data Structures and Networks are not directly related in WaMDaM. They are related through the instances of an Object Types that belong to a Data Structure. Those instances belong to a Network. Users will use this relationship to query all the networks that belong to a Data Structure. A software business rule must enforce that a Network must have at least one node instance to serve as the connection with the Data Structure. So the selected Data Structure in **Figure 13** will be used in the sub-steps of importing nodes and links to connect the Data Structure with its Network.

All instances of the same Object Type must inherit the same set of attributes since instances are the children of ObjectTypes. The logical data model of WaMDaM does not have explicit and embedded business rules to implement this important feature in the database. WaMDaM allows instances of the same Object Type to inherit one or more of the Object Types attributes, not all of them. Therefore, the Wizard must add a software business rule for this feature as the following explains.

The Table ObjectAttributes serves a purpose to link Instances with their Attributes and their Object Types (**Figure 17**). When a user creates the first Object Type in the database, the Wizard automatically creates a new attribute called "ObjectInstances" in the Attributes Table (unless it already exists). Then the Wizard automatically creates a new ObjectAttribute entry that assigns the new added ObjectType with the Attribute "ObjectAttributeID". For each additional added Object Type, the Wizard will automatically create a new ObjectAttribute entry that reuses the Attribute "ObjectInstances" to make the connection in **Figure 17** .

When a new instance is added, an Objet Type must be provided with it as indicated by the spreadsheet template. The Wizard will use the provided Object Type and the "ObjectInstances" attribute to look up the corresponding ObjectAttributeID. The same ObjectAttributeID will be inserted in the Mapping Table with each instance with the same Object Type. This ObjectAttributeID will connect the ObjectType and its Attributes with its Instance in the database.

Each added instance that has the same ObjectAttributeID will inherit all the attributes of the same Object Type. Then, the Mapping table will be populated with all the Object Attributes for each new instance. At the same time, once a new entry is added to the Mapping Table, the Wizard automatically creates a new entry in the ScenarioMapping Table that connects the instance, with an attribute, and a scenario. The ScenarioMapping table helps sharing data between scenarios without duplications. Only unique data differences between scenarios will be different between two scenarios.

The Wizard looks up the SourceID and MethodID for each new instances and its attribute "ObjectInstances" based on the provided source name and method in the Nodes and Links spreadsheets which tells about obtaining the instances itself. When populating the Mapping Table for the rest of attributes of an ObjectType for each instance, the Wizard will allow Nulls for the SourceID and MethodID. When the user later loads data values in Section 3.3, the identifiers will be required to populate the Mapping Table in the second time based on the provided source and method names for data values. The last field in the Mapping Table is DataStorageID which is a unique sequential identifier for each data value that connects it with the chain of metadata up to the DataStructure. There are two choices on when to create the DataStorageID and update it in the Mapping Table. The first is to create it when the user loads a new instance. The second is to create the Data Storage record when the user loads new data values. The Wizard can apply either choice based on which one is easier to program.

The AttributeTypeCodeCV as defined for each ObjectAttribute will specify the data type for each attribute which will be implemented to all instance of the same Object Type. The Wizard needs to implement a software business rule that allows users to only populate one of the eight data types as chosen in the ObjectAttribute table. Next I explain the third major step to import data into WaMDaM for the defined Data Structure and Network, which is to import Data Values.

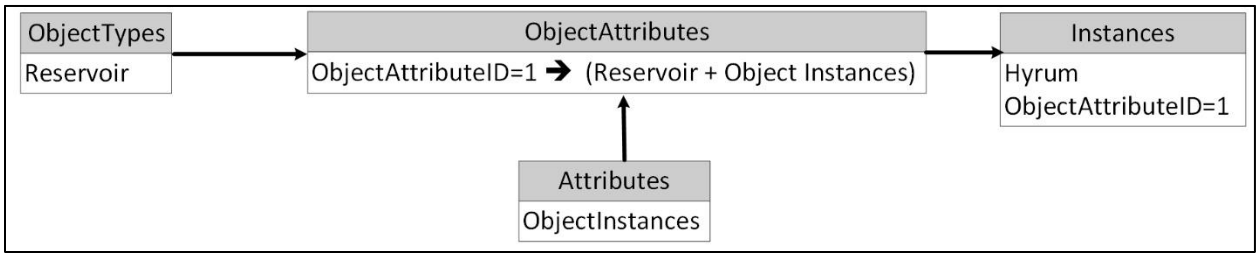


Figure 17: An illustration of how the Object Attributes Table connects or bridges between Object Type, Instances, and Attributes.

3.3 Import Data Values from the spreadsheet to the WaMDaM database

Once the user has defined a network and a scenario which include nodes and links in Sections 3.1 and 3.2, they can import data values that describe the attributes of the nodes and links. There are eight different independent data types that can describe an attribute (**Figure 18**). The user has to first organize the network data values based on their type in separate spreadsheets in the Data Values workbook. Next, users can import the data values for all the attributes of all the nodes and links of the network in one time. As mentioned earlier Section 3.2, the DataStorageID is unique for each combination of an attribute, instance, scenario, source, and method and it connects all of them with a data value in any of the nine Tables. The DataStorage bridge table allows "many to many" relationships between the combination and data values. A combination can have many data "text controlled" values like many purpose values for a reservoir instance, "recreation", flood control", and "hydropower". At the same time "recreation" text controlled value can be shared across many reservoir instances like Hyrum and Cutler reservoirs.

In the workbook template, the user will provide the above metadata as text values. The scenario will be selected in the Wizard which applies to all the binary imported values. The Wizard has to look up the primary key for each metadata value and search for the unique match of all of them to end up with the single key of the DataStorageID. The Storage ID will be used as a foreign key in each of the Data Values Tables. Once the user click on a button for a data type to import, a window pops up (**Figure 19**) which is similar to all the data types. The Time Series and Multi-Column Arrays are a bit trickery to load data to because each of them has two tables. The first table includes general metadata that can be applied to the second table.

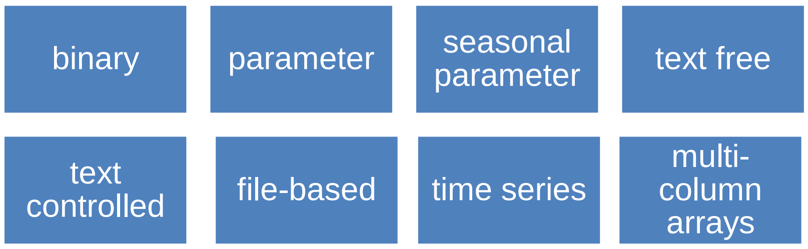
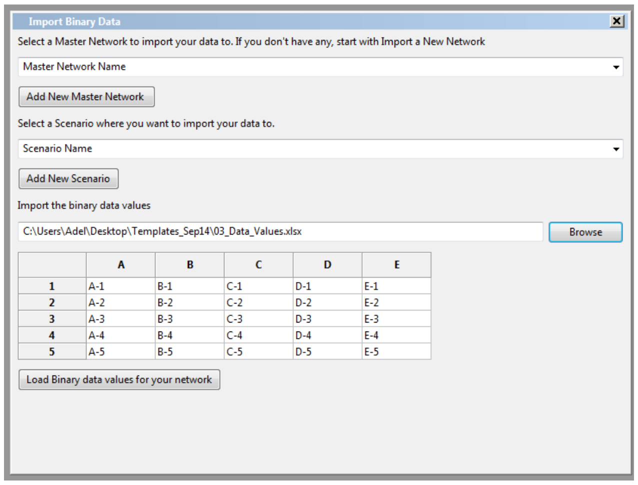


Figure 18: The eight independent steps to import data values to WaMDaM



Select a Master Network to import your data to. If you don't have any, start with Import a New Network

Select a Scenario where you want to import your data to.

Import the binary data values

C:\Users\Adel\Desktop\Templates_Sep14\03_Data_Values.xlsx

	A	B	C	D	E
1	A-1	B-1	C-1	D-1	E-1
2	A-2	B-2	C-2	D-2	E-2
3	A-3	B-3	C-3	D-3	E-3
4	A-4	B-4	C-4	D-4	E-4
5	A-5	B-5	C-5	D-5	E-5

Figure 19: Import Binary Data Values

4. Import metadata and controlled vocabularies to the WaMDaM Database

Within the three major steps mentioned above in Sections 3.1-3.3 to import data into WaMDaM, the user will be prompted to define new metadata and controlled vocabulary entries that do not exist in the database. If the user chooses to add a new CV term or metadata, a pop up window will guide them to do so and import metadata or CVs from spreadsheets as explained in the next two sub-sections 4.1 and 4.2.

4.1 Import Metadata

There are five metadata tables in the WaMDaM Wizard (**Figure 20)**. Both Object Category and Attribute Category are independent tables and they are optional in WaMDaM. “Sources” and “Methods” metadata tables depend on “People”, and the three depend on “Organizations”. When the user clicks at “Organizations”, the dialog box in **Figure 21** pops up. The user navigates to the workbook of “Metadata”. The Wizard automatically looks into the sheet name “Organizations”. A pop up window views the data as inserted to the sheet. Then the user can click the button “load organizations data”. The Wizard will run validation checks to see if the data entries comply with the physical data types and required fields. The user follow similar steps for importing People, Sources, Methods, Object Category, and Attribute Category. Importing “people” data depends on importing “Organizations” and importing either Methods and or Sources depend on importing “People”. The buttons to Sources and Methods will be inactive until the table “people” is populated. Similarly, the “People” button will be inactive until the “Organizations” table is populated. If a user tries to import data that depends on another table but no matching table exist in the independent table, an error message will inform the user to load the independent table first.

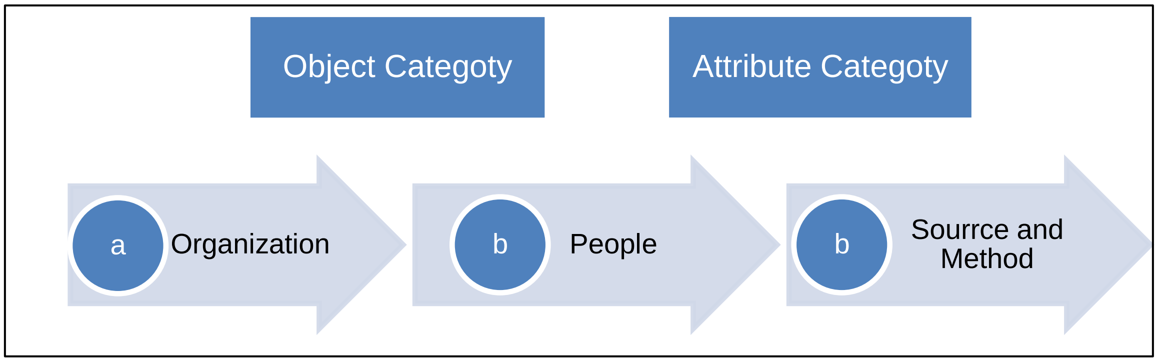
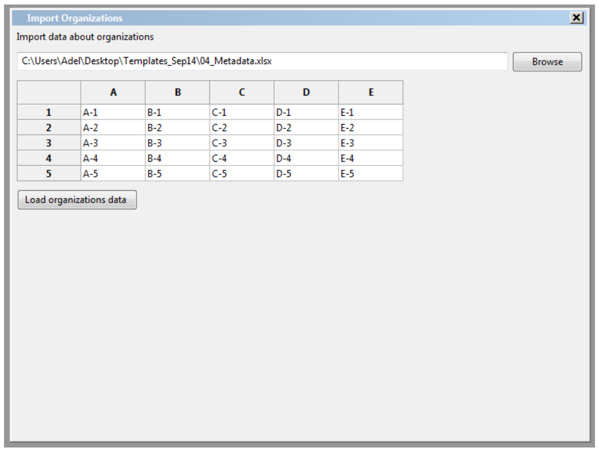


Figure 20: Metadata tables and the consecutive steps to populate dependent tables in WaMDaM



Import data about organizations

C:\Users\Adel\Desktop\Templates_Sep14\04_Metadata.xlsx

	A	B	C	D	E
1	A-1	B-1	C-1	D-1	E-1
2	A-2	B-2	C-2	D-2	E-2
3	A-3	B-3	C-3	D-3	E-3
4	A-4	B-4	C-4	D-4	E-4
5	A-5	B-5	C-5	D-5	E-5

Figure 21: Import “Organizations” dialog window

4.2 Import Controlled Vocabularies (CVs)

There are fourteen controlled vocabulary tables supported in the WaMDaM Wizard. Two depend on other tables while the rest of tables are independent (**Figure 22**). WaMDaM will be populated with controlled vocabularies and users have the choice to import more (**Figure 23**).

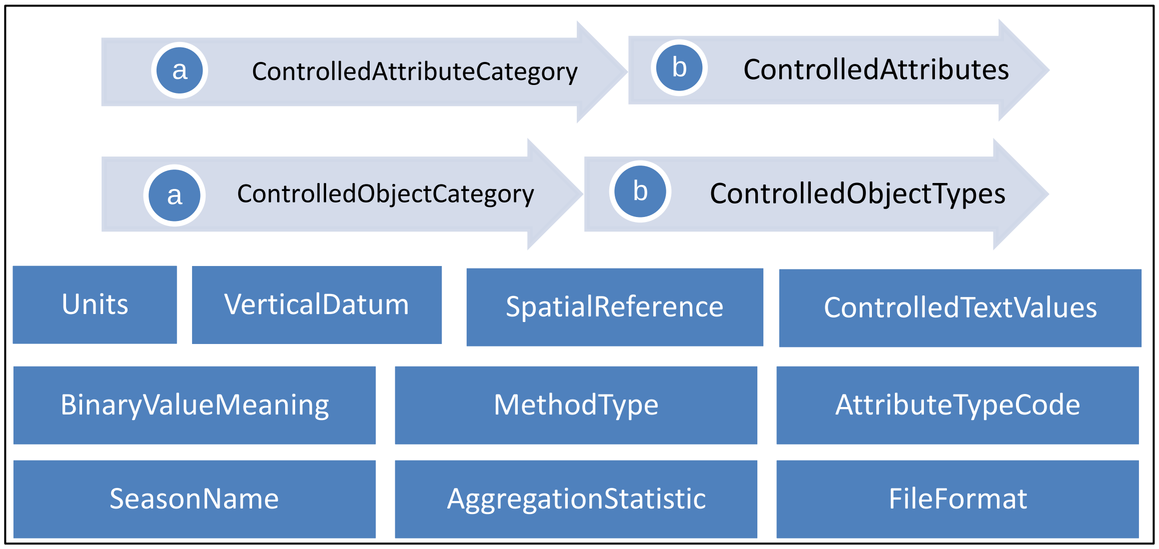
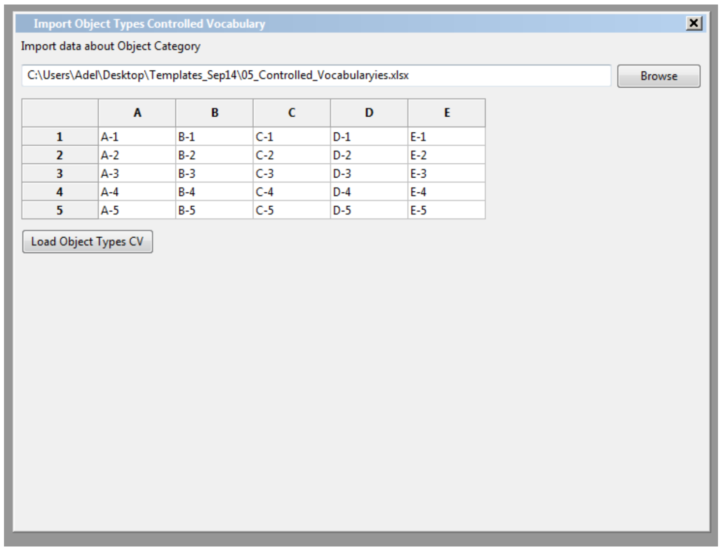


Figure 22: Controlled vocabularies in WaMDaM Wizard


Import Object Types Controlled Vocabulary

Import data about Object Category

C:\Users\Adel\Desktop\Templates_Sep14\05_Controlled_Vocabularyies.xlsx

	A	B	C	D	E
1	A-1	B-1	C-1	D-1	E-1
2	A-2	B-2	C-2	D-2	E-2
3	A-3	B-3	C-3	D-3	E-3
4	A-4	B-4	C-4	D-4	E-4
5	A-5	B-5	C-5	D-5	E-5

Figure 23: Importing Object Types CVs in WaMDaM Wizard

5. About WaMDaM

The “About” tab in the Wizard homepage has three buttons (
Figure 24). Clicking on “About WaMDaM” bring a window with a summary about it (**Figure 25**). Clicking on “License” bring a window with the full license details (**Figure 26**). Clicking on “help” button bring a window with descriptions/workflow that explain how someone can use the Wizard (**Figure 27**).

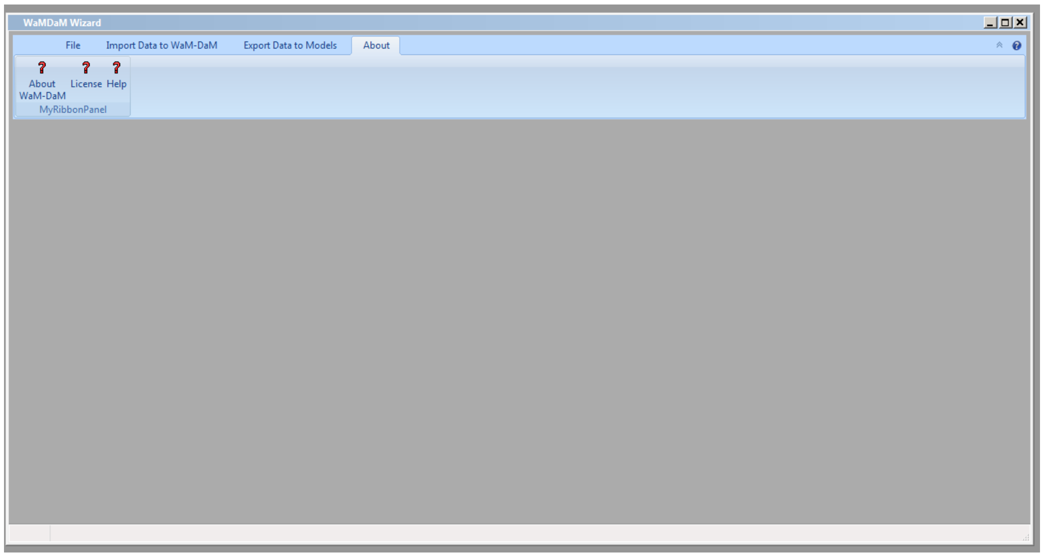
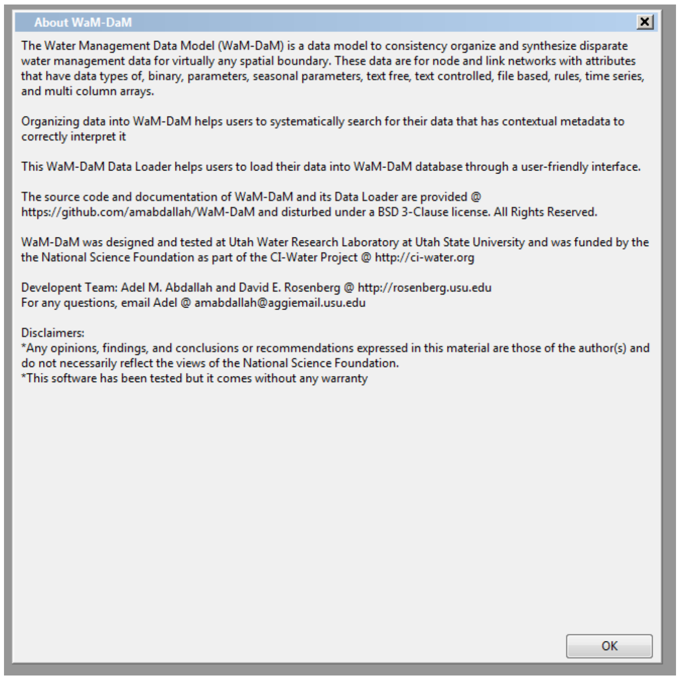


Figure 24: The Wizard homepage window for “About


About WaM-DaM

The Water Management Data Model (WaM-DaM) is a data model to consistency organize and synthesize disparate water management data for virtually any spatial boundary. These data are for node and link networks with attributes that have data types of, binary, parameters, seasonal parameters, text free, text controlled, file based, rules, time series, and multi column arrays.

Organizing data into WaM-DaM helps users to systematically search for their data that has contextual metadata to correctly interpret it

This WaM-DaM Data Loader helps users to load their data into WaM-DaM database through a user-friendly interface.

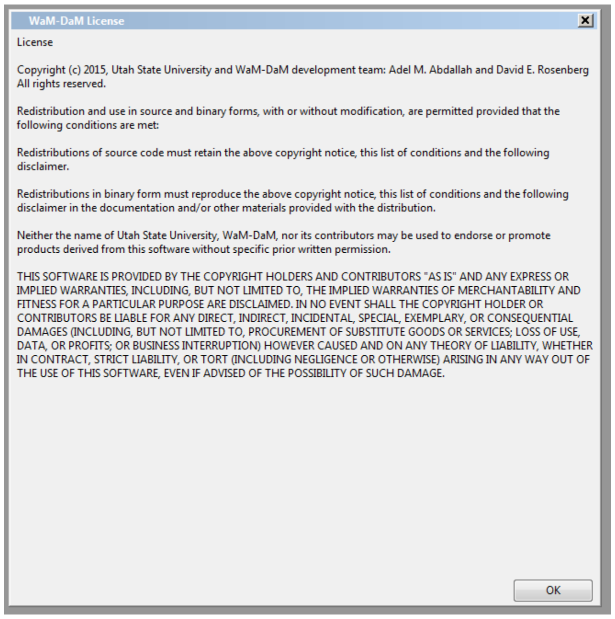
The source code and documentation of WaM-DaM and its Data Loader are provided @ https://github.com/amabdallah/WaM-DaM and disturbed under a BSD 3-Clause license. All Rights Reserved.

WaM-DaM was designed and tested at Utah Water Research Laboratory at Utah State University and was funded by the the National Science Foundation as part of the CI-Water Project @ http://ci-water.org

Developent Team: Adel M. Abdallah and David E. Rosenberg @ http://rosenberg.usu.edu
For any questions, email Adel @ amabdallah@aggiemail.usu.edu

Disclaimers:
*Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.
*This software has been tested but it comes without any warranty

Figure 25: dialog box of “About WaMDaM



License

Copyright (c) 2015, Utah State University and WaM-DaM development team: Adel M. Abdallah and David E. Rosenberg
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

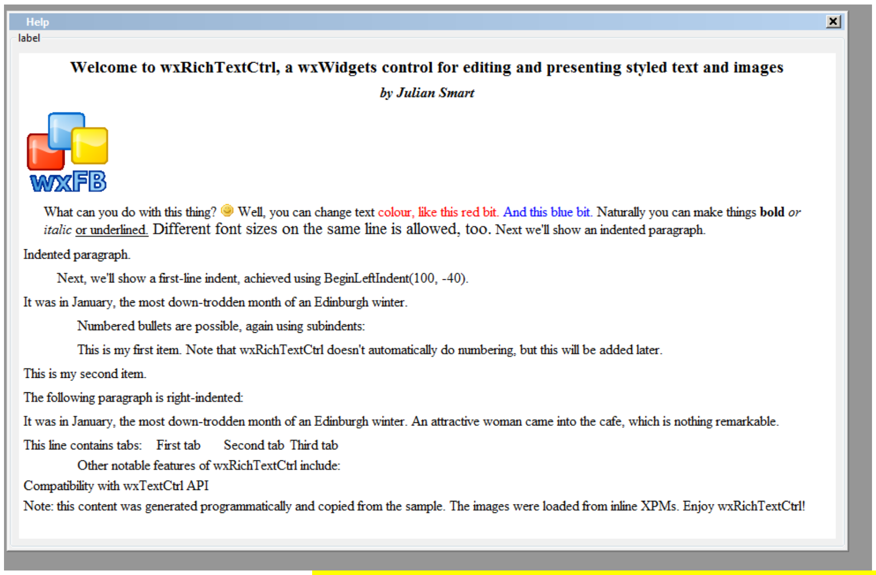
Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

Neither the name of Utah State University, WaM-DaM, nor its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Figure 26: WaMDaM dialog box of for the license



Welcome to wxRichTextCtrl, a wxWidgets control for editing and presenting styled text and images

by Julian Smart

What can you do with this thing? Well, you can change text colour, like this red bit. And this blue bit. Naturally you can make things **bold** *or italic* or underlined. Different font sizes on the same line is allowed, too. Next we'll show an indented paragraph.

Indented paragraph.

Next, we'll show a first-line indent, achieved using BeginLeftIndent(100, -40).

It was in January, the most down-trodden month of an Edinburgh winter.

Numbered bullets are possible, again using subindents:

This is my first item. Note that wxRichTextCtrl doesn't automatically do numbering, but this will be added later.

This is my second item.

The following paragraph is right-indented:

It was in January, the most down-trodden month of an Edinburgh winter. An attractive woman came into the cafe, which is nothing remarkable.

This line contains tabs: First tab Second tab Third tab

Other notable features of wxRichTextCtrl include:

Compatibility with wxTextCtrl API

Note: this content was generated programmatically and copied from the sample. The images were loaded from inline XPMs. Enjoy wxRichTextCtrl!

Figure 27: Dialog box of “Help”. The text will define the steps and their hierarchy to import data into WaMDaM. I cannot figure out how to edit this rich text in wxFormBuilder

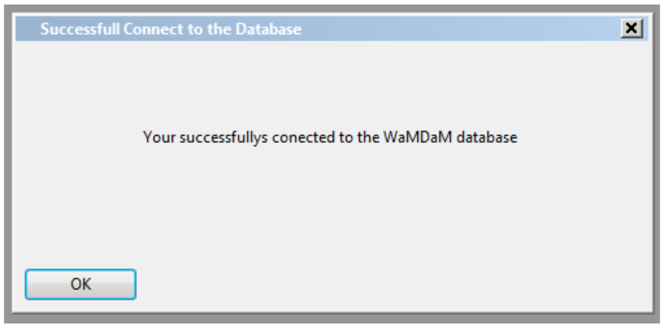
6. Data Validation, integrity checks, and transaction management

WaMDaM Wizard will implement three levels of validation on data that are loaded from any spreadsheet to insure that the integrity of the data is maintained, these levels are: i) read and verify the spreadsheet entries, ii) insert data values to the database, iii) register entries against metadata and controlled vocabularies. A message must show up and tell the user if their action to connect to the database or load data is successful (**Figure 28**). An error message must show up for each error that violates any validation check (**Figure 29**). For each error, the message should be specific and mention the name of the spreadsheet or attribute that violates the rule so the user can fix it as easy as possible. The error message should be understood by users without the need to look up the logical design.

In the first validation level, upon reading the file and loading it into WaMDaM Wizard, it will be parsed and checked for consistency with WaMDaM requirements and constraints. This includes checking for spreadsheet name and headers. Although the workbooks sheet titles and headers will be locked in Excel, this validation check screens out any other spreadsheets that the user might choose by mistake. Using the same spreadsheet names as provided in each template is important because the Wizard will automatically locate the spreadsheet inside the workbook for each step base on its predefined name. Once input data in the selected spreadsheet has been validated at this level, it will be passed to the database for loading.

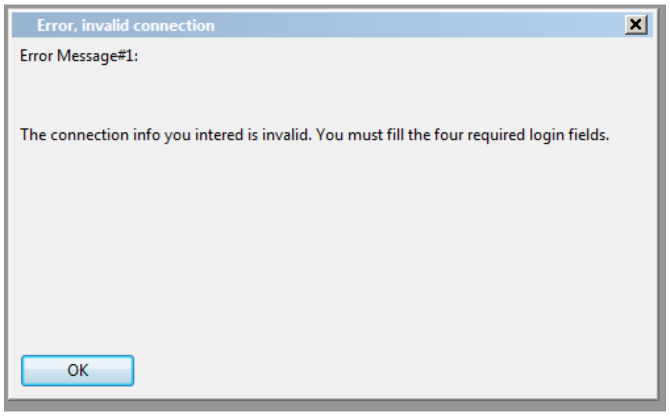
Second, validation occurs when WaMDaM Wizard tries to insert the data into the database (i.e., the WaMDaM database will apply all of its constraints). Check of the physical data type for each data entry matches the attribute data type (e.g., numeric, text, and date), required fields, fields cannot be null, fields that do not allow special characters. It is possible to implement these validations in the spreadsheet itself through macros. However, the validations in the spreadsheet might be frustrating to users in case they want to use the spreadsheet at different sessions like to populate half of the data and come back to the second half later. WaMDaM Wizard will check to make sure that records added to each table within WaMDaM are unique so that duplicates are not loaded into the database. Finally the wizard will check for dependency on other tables through foreign keys. For example if the user wants to add a time series spreadsheet but the network has not been defined, then the user gets an error message that asks the user to first define a network.

In the third level, users will have the option to define and register data values of a number of field names against controlled vocabularies that they add or use the existing ones. When the Wizard tries to load data entries to the database, the Wizard will try to match and register the new loaded entries for specific list of attributes with the existing controlled vocabularies. If both terms do not fully match, then the Wizard will prompt the user to choose to where to register the data value against a list of suggested relevant terms in the CV. The Wizard will facilitate these options to the users: ignore registering CVs, the provided term matches an existing CV term, suggest matching entries with controlled vocabulary, define new controlled vocabulary term by showing the import dialog box from spreadsheet for the specific CV at hand. The following section discusses the software architecture of the Wizard.


Successfull Connect to the Database

Your successfullys conected to the WaMDaM database

Figure 28: An example message for a successful action in the Wizard


Error, invalid connection

Error Message#1:

The connection info you intered is invalid. You must fill the four required login fields.

Figure 29: An example message for an error in the Wizard

7. Architecture

The software of the WaMDaM Wizard will use the Model–View–Controller (MVC) architecture (**Figure 30**). The Model represents the data access layer and the select, update queries to database. The View represents the GUI presentation with buttons and renders the Model data. The Controller represents the business logic that defines the application behavior based on the user interactions in the View layer. Separating the three layers from each other allows for flexibility to update each layer without major changes in the other layers. The wxPython script must not be written around the GUI Forms built by the wxFormBuilder. The script should import the GUI forms so that any cosmetic changes in Forms shall be updated without affecting any of the script. For example, updating the “look and feel” of the GUI files should not affect the rest of the Wizard functionality at all. The wxPython script must be well commented for each file and step taken.

The Wizard script must use the consistent naming conversion in

Table *1*. This conversion allows programmers to search wxPython based on the beginning letters that narrow down the search to the item type. Additional items that do not exist in should be added to the Table following similar style. The WaMDaM Wizard application and its source code will be made freely available on GitHub according to the BSD 3-Clause license.

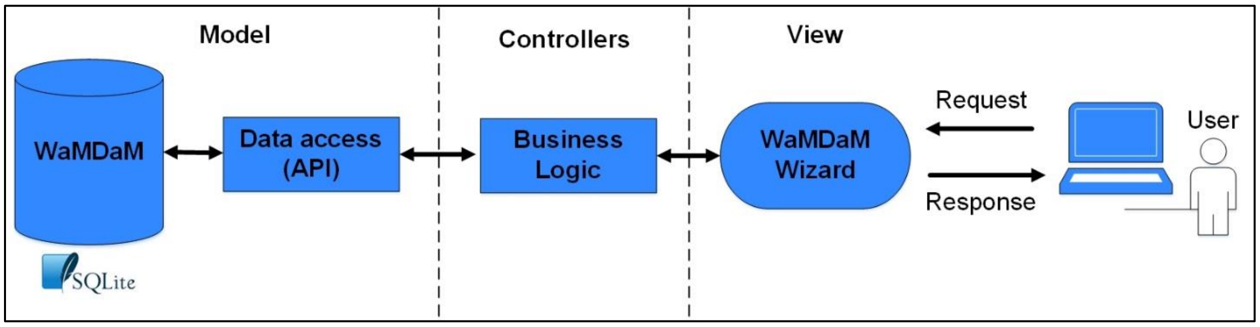


Figure 30: Model-Controller-View architecture of the WaMDaM Wizard software

Table 1: Wizard script naming conversions. The name comes out after the underscore "_" in the abbreviated name

Wizard item Name	Abbreviated name
Button	btn_
Wizard	wiz_
Dialog window	dilg_
Frame	frm_
Combo Box	comboBox_
File Picker	FileBrows_
Text Control	textCtrl_
Wizard page	wizPag_
Add more if you see them	

Disclaimers

The development of the Water Management Data Model (WaMDaM) was part of the Cyber Infrastructure (CI-Water) Project to advance water resources modeling funded by the National Science Foundation Grant No. EPS-1135482. Utah Water Research Lab provided funding as well. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the National Science Foundation. Although much effort has been expended in the development and testing of the WaMDaM Wizard application, errors and may still occur as the BSD 3-Clause license implies.

Acknowledgements

The team of developers that contributed to WaMDaM Wizard includes: Adel M. Abdallah and David E. Rosenberg. We thank Stephanie Reeder at Utah Water Research Lab for her support.

Technical Support

There will be no formal ongoing support for this freely distributed open source software. However, we are interested in feedback. If you find errors, have suggestions, or are interested in any later versions, please contact:

Adel M. Abdallah
Utah State University
8200 Old Main Hill
Logan, UT 84322-8200
amabdallah@aggiemail.usu.edu

License