Sample for the Simple Script Wrapper

# Overview

The Simple Script Wrapper (SSW) is an Open Modeling Interface (OpenMI) component that makes it possible to link scripts written in MATLAB, Scilab, and Python to other OpenMI components. See the journal article for complete details[[1]](#footnote-1). This document presents a sample that demonstrates how the SSW may be used.

The sample consists of 3 models, each written in a different scripting language. The models are simplifications meant only for demonstration. The models represent weather, crop production, and groundwater. The included composition file Composition.opr is shown in Fig. 1 as it appears inside the Configuration Editor.

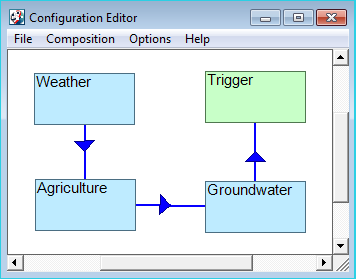


Figure 1. The models as they appear in the Configuration Editor.

The weather model is implemented in Python and generates random precipitation data that is provided to the agricultural model. The agricultural model is implemented in MATLAB and calculates crop water consumption based on a constant need and variable precipitation. The agricultural model calculates the groundwater pumping rate as the difference between the crop water need and the precipitation received. This pumping rate is provided to the groundwater model. The groundwater model is written in Scilab and calculates the total groundwater storage.

The models are configured to operate on an annual time step over a 4-county area (i.e. 4 elements in the element set). Sample output from the linked model is shown in Fig. 2.

Macintosh HD:Users:tombz:Desktop:Output.pdf

Figure 2. Sample output from the linked model for 4 counties.

# System Requirements

To run the sample linked model, the following software is required: OpenMI Configuration Editor 1.4, MATLAB R2010, Scilab 5.3.3, and Python 2.7. If these applications are not installed to their default locations, it will be necessary to specify the install path in the Component.omi files, for example:

<Argument Key="RuntimePath" ReadOnly="true" Value="C:\Program Files\scilab-5.3.3\bin" />

<Argument Key="RuntimePath" ReadOnly="true" Value="C:\Program Files\MATLAB\R2010a\bin\win64" />

<Argument Key="RuntimePath" ReadOnly="true" Value="C:\Python27" />

# How It Works

Each model consists of 3 core functions: sswInitialize, sswPerformTimeStep, and sswFinish. For MATLAB and Scilab each of these is in a separate script file and for Python they are combined into a single Model.py file. These functions contain the scripting statements that setup the model, perform a single time step, and clean up. Global variables can be shared among the functions (although be sure to use different global variable names if you are using multiple Scilab models within a single composition).

The 3 functions obtain the input from other models, and provide output to other models (as well as obtain state information such as the current simulation time) by calling a special set of SSW functions shown in Table 1.

Table 1. Script function API provided by the SSW.

|  |  |
| --- | --- |
| Function | Description |
| sswGetInput(quantityName) | Returns values from other components |
| sswSetOutput(quantityName, value) | Saves values for other components |
| sswGetCurrentTime() | Returns the current simulation time |
| sswGetScriptFolder() | Returns the path to the script folder |

The quantityName is the name of the quantity as it appears in the Component.xml file. The Component.xml file contains metadata about the model, such as the time span that it can simulate and the list of input and output exchange quantities.

The sswPerformTimeStep() function typically has 3 parts that (1) call sswGetInput() to obtain inputs from other components, (2) calculates new values of its state variables for the current simulation time, and then (3) calls sswSetOutput() to save the new state variable values so that they will be available to other components.

The API script functions in Table 1 are automatically generated by the SSW when a model script is loaded into the Configuration Editor. When writing and testing a model, it is helpful to be able to run the model by itself, outside of the Configuration Editor. This is possible through the use of placeholder functions that take the place of the API functions. Each of the models has a set of placeholder functions in the Solo folder of each model, and the Solo script file can be executed to run the model from the console of MATLAB, Scilab, or Python. The Solo script mimics the way that the SSW calls the 3 core functions.

1. http://dx.doi.org/10.1016/j.envsoft.2012.07.006 [↑](#footnote-ref-1)